Journal of Computer and System Sciences 74 (2008) 1309–1331

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

An analysis of model-based Interval Estimation for Markov Decision Processes

Alexander L. Strehl [a], Michael L. Littman [b,*]

[a] Yahoo! Inc, 701 First Avenue, Sunnyvale, California 94089, USA
[b] Computer Science Department, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854, USA

ARTICLE INFO

Article history:
Received 22 August 2007
Available online 19 September 2008

Keywords:
Reinforcement learning
Learning theory
Markov Decision Processes

ABSTRACT

Several algorithms for learning near-optimal policies in Markov Decision Processes have been analyzed and proven efficient. Empirical results have suggested that Model-based Interval Estimation (MBIE) learns efficiently in practice, effectively balancing exploration and exploitation. This paper presents a theoretical analysis of MBIE and a new variation called MBIE-EB, proving their efficiency even under worst-case conditions. The paper also introduces a new performance metric, average loss, and relates it to its less "online" cousins from the literature.

1. Introduction

In the reinforcement-learning (RL) problem [21], an agent acts in an unknown or incompletely known environment with the goal of maximizing an external reward signal. One of the fundamental obstacles in RL is the exploration–exploitation dilemma: whether to act to gain new information (explore) or to act consistently with past experience to maximize reward (exploit). This paper deals with the RL problem as modeled using a general Markov Decision Process (MDP) environment with finite state and action spaces.

Interval Estimation (IE) was introduced as an approach to the exploration problem in the simpler *k-armed bandit problem* [13,9]. There have been several attempts to generalize IE to MDPs [9,24]. Although these early studies made some questionable statistical assumptions[1] and proved no formal results, they did perform extensive empirical comparisons that demonstrated IE as a noteworthy technique for handling exploration in MDPs. Additional promising empirical results were also presented by Strehl and Littman [19]. We introduce a version of *Model-based Interval Estimation* (*MBIE*), which not only combines Interval Estimation with model-based reinforcement learning, but also comes with a formal PAC-like[2] learning-time guarantee in MDPs. We fully develop and significantly analyze this algorithm. Preliminary results of our analysis appear in [20]. The current work expands the analysis and corrects some errors that appear there.

Along the way, we survey several different reinforcement-learning performance metrics that have a PAC-like intuition. We conclude with a new and significantly more "online" metric called *average loss* and formally relate it to a standard metric.

* Corresponding author.
E-mail addresses: strehl@yahoo-inc.com (A.L. Strehl), mlittman@cs.rutgers.edu (M.L. Littman).

[1] Both used confidence intervals derived from the normal distribution.

[2] PAC stands for "Probably Approximately Correct."

0022-0000/$ – see front matter
doi:10.1016/j.jcss.2007.08.009

1.1. General form of Interval Estimation

The Interval Estimation approach is easily understand when framed in terms of the *k-armed bandit problem*, which provides a minimal mathematical model in which to study the exploration–exploitation tradeoff. Here, we imagine we are faced with a slot-machine with a set of k arms, only one of which can be selected on each timestep. When selected, each arm i produces a single payoff drawn from a stationary but unknown payoff distribution with finite mean μ_i and variance.

The optimal long-term strategy for a k-armed bandit is to always select the arm with the highest mean payoff, $\text{argmax}_i\, \mu_i$. However, these expected payoffs are not known to the agent, who must expend a number of trials in exploration. By the law of large numbers, increasingly accurate estimates of the mean payoff of each arm can be made by repeated sampling. However, an agent that continues to sample all arms indefinitely will be wasting opportunities to exploit what it has learned to maximize its payoffs.

The general form of the Interval Estimation (IE) strategy is as follows.

- On the nth trial, construct a δ_n% confidence interval for the mean of the payoff distribution for each arm. For arm i, the confidence interval will be of the form $[\hat{\mu}_i - \epsilon_i, \hat{\mu}_i + \epsilon_i]$, where $\hat{\mu}_i$ is the average of the payoffs received so far from choosing arm i and ϵ_i is a real number measuring the confidence of $\hat{\mu}_i$ as an estimate to the true mean payoff μ_i. An arm whose confidence has the highest upper tail is then chosen. Formally, we choose arm j such that $j = \text{argmax}\{\hat{\mu}_i + \epsilon_i \mid i = 1, \ldots, n\}$.

Different implementations of IE must choose, at each step, concrete values for δ_n and ϵ_i. The IE algorithm exhibits a well-principled approach to exploration. Suppose the confidence intervals are admissible, meaning each contains its true mean payoff ($\mu_i \in [\hat{\mu}_i - \epsilon_i, \hat{\mu}_i + \epsilon_i]$). If the confidence interval for the chosen arm is tight, meaning it consists only of similar values, then its mean payoff is very close to the maximum mean payoff of any arm. Otherwise, if the confidence interval is very loose, then the mean payoff of the selected arm may or may not be near-optimal. Thus, it makes sense to obtain more information about its behavior.

Next, we define the main learning framework studied in the rest of the paper.

1.2. Markov Decision Processes

This section introduces the Markov Decision Process (MDP) notation used throughout the paper; see [21] for an introduction. Let $\mathcal{P}_S$ denote the set of all probability distributions over the set S. A finite MDP M is a five tuple $\langle S, A, T, \mathcal{R}, \gamma \rangle$, where S is a finite set called the state space, A is a finite set called the action space, $T : S \times A \to \mathcal{P}_S$ is the transition distribution, $\mathcal{R} : S \times A \to \mathcal{P}_{\mathbb{R}}$ is the reward distribution, and $0 \leqslant \gamma < 1$ is a discount factor on the summed sequence of rewards. We call the elements of S and A states and actions, respectively. We let $T(s'|s,a)$ denote the transition probability of state s' of the distribution $T(s,a)$. In addition, $R(s,a)$ denotes the expected value of the distribution $\mathcal{R}(s,a)$.

We assume that the learner (also called agent) receives S, A, ϵ, δ, and γ as input. The learning problem is defined as follows. The agent always occupies a single state s of the MDP M. The agent is told this state and must compute an action a. It then receives an *immediate reward* $r \sim \mathcal{R}(s,a)$ and is transported to a *next-state* $s' \sim T(s,a)$ according to the rules from above. This procedure then repeats forever. The first state occupied by the agent may be chosen arbitrarily. We discuss various ways of evaluating an agent's performance in Section 2. We define a *timestep* to be a single interaction with the environment, as described above. We also define an *experience* of state-action pair (s,a) to refer to the event of taking action a from state s. The real-valued quantities ϵ and δ control the behavior of the algorithm. We will be interested in algorithms that satisfy Definition 2 in Section 2.

A *policy* is any strategy for choosing actions. A stationary policy is one that produces an action based on only the current state. We assume (unless noted otherwise) that rewards all lie between 0 and 1. For any policy π, let $V_M^\pi(s)$ ($Q_M^\pi(s,a)$) denote the discounted, infinite-horizon value (action-value) function for π in M (which may be omitted from the notation) from state s. If H is a positive integer, let $V_M^\pi(s,H)$ denote the H-step value function of policy π. If π is non-stationary, then s is replaced by a *partial path* $c_t = s_1, a_1, r_1, \ldots, s_t$, in the previous definitions. Specifically, let s_t and r_t be the tth encountered state and received reward, respectively, resulting from execution of policy π in some MDP M. Then, $V_M^\pi(c_t) = E[\sum_{j=0}^{\infty} \gamma^j r_{t+j}]$ and $V_M^\pi(c_t, H) = E[\sum_{j=0}^{H-1} \gamma^j r_{t+j}]$. These expectations are taken over all possible (in)finite paths the agent might follow. The optimal policy is denoted π^* and has value functions $V_M^*(s)$ and $Q_M^*(s,a)$. Note that a policy cannot have a value greater than $1/(1-\gamma)$.

1.3. Overview of main results

Our main result is to prove that a version of MBIE is PAC-MDP in general MDPs (Theorem 1). Thus, with high probability, it will learn quickly in even the hardest of MDPs. We also prove that a simplifying variation of MBIE, called MBIE-EB (MBIE with exploratory bonus) is also PAC-MDP (Theorem 2). This provides a formal validation of the learning efficiency of MBIE. The new algorithm, MBIE-EB, is also very attractive in terms of its simplicity, easiness of implementation, and computational complexity.

Another noteworthy result is that a PAC-MDP algorithm in the sample complexity framework (see Section 2) is also PAC-MDP in the average loss setting (Section 5). A guarantee of an algorithm's average loss is highly desirable because it provides

a guarantee on the actual return (discounted sum of rewards) the agent will receive compared to the actual return of an optimal agent. Our result thus validates the use of sample complexity as a performance metric; small sample complexity implies low average loss.

1.4. Related work

Various forms of the Interval Estimation algorithm for the k-armed bandit problem have been studied. It has generally been shown to perform well empirically [9,7] and theoretically [1,16].

Past attempts to generalize IE to learning in MDPs include [9,24]. A similar idea has been explored in the Bayesian setting [25]. Early empirical results indicating MBIE's learning efficiency can be found in [19]. The work of Even-Dar et al. [5] also investigates the use of confidence intervals in the MDP setting. Rather than choosing the action, a^*, whose confidence interval has maximum tail, Action Elimination chooses arbitrarily among those actions whose confidence intervals overlap with the confidence interval of a^*. Our work builds upon all of these previous works in that we analyze the learning complexity of a version of MBIE and show that it satisfies a PAC-like learning guarantee. Similar analyses of other reinforcement learning algorithms include [6,11,3,10].

Recently, Strehl et al. [17] have shown that two PAC-MDP model-based algorithms (one based on R-max and another on MBIE-EB) can be modified so that they do not solve their model completely during each timestep. They show that completely removing the planning step greatly reduces each algorithm's per-step computational complexity with little asymptotic change to their sample complexity. In addition, Strehl et al. [18] provide a model-free algorithm called *Delayed Q-learning* that is PAC-MDP and has a resemblance to Q-learning. Surprisingly, the sample complexity bounds for Delayed Q-learning improve on all previous bounds when only the dependence on $|S|$ and $|A|$ are considered. Practically, however, MBIE and MBIE-EB greatly outperform the previously mentioned algorithms, in terms of maximizing total or discounted return.

Finally, we mention that Auer and Ortner [2] present an algorithm similar to MBIE and show that it achieves *logarithmic regret*. We also note that the PAC-MDP framework is inherently different from the regret framework, although both are interesting. In the PAC-MDP framework, an algorithm is required to quickly learn near-optimal behavior, with high probability. To obtain sublinear regret, however, an algorithm's behavior must be able to achieve performance arbitrarily close to that of the optimal policy. On the other hand, the result of Auer and Ortner [2] is for expected regret while our results are with high probability (which is stronger). In addition, our PAC-MDP sample complexity bounds are on the order of S^2A, when logarithmic factors are ignored, but the corresponding terms in the regret analysis are much higher.

2. Performance metrics

A reasonable notion of learning efficiency in an MDP is to require an efficient algorithm to achieve near-optimal (expected) performance with high probability. An algorithm that satisfies such a condition can generally be said to be *probably approximately correct* (PAC) for MDPs. The PAC notion was originally developed in the supervised learning community, where a classifier, while learning, does not influence the distribution of training instances it receives [22]. In reinforcement learning, learning and behaving are intertwined, with the decisions made during learning profoundly affecting the available experience.

In applying the PAC notion in the reinforcement-learning setting, researchers have examined definitions that vary in the degree to which the natural mixing of learning and evaluation is restricted for the sake of analytic tractability. We survey these notions next.

One difficulty in comparing reinforcement-learning algorithms is that decisions made early in learning can affect significantly the rewards available later. As an extreme example, imagine that the first action choice causes a transition to one of two disjoint state spaces, one with generally large rewards and one with generally small rewards. To avoid unfairly penalizing learners that make the wrong arbitrary first choice, Fiechter [6] explored a set of PAC-learning definitions that assumed that learning is conducted in trials of constant length from a fixed start state. Under this *reset assumption*, the task of the learner is to find a near-optimal policy from the start state given repeated visits to this state.

Fiechter's notion of PAC reinforcement-learning algorithms is extremely attractive because it is very simple, intuitive, and fits nicely with the original PAC definition. However, the assumption of a reset is not present in the most natural reinforcement learning problem. Theoretically, the reset model is stronger (less general) than the standard reinforcement learning model. For example, in the reset model it is possible to find arbitrarily good policies, with high probability, after a number of experiences that does not depend on the size of the state space. However, this is not possible in general when no reset is available [10].

Kearns and Singh [11] provided an algorithm, E^3, which was proven to obtain near-optimal return quickly in both the average reward and discounted reward settings, without a reset assumption. Kearns and Singh note that care must be taken when defining an optimality criterion for discounted MDPs. One possible goal is to achieve near-optimal return from the initial state. However, this goal cannot be achieved because discounting makes it impossible for the learner to recover from early mistakes, which are inevitable given that the environment is initially unknown. Another possible goal is to obtain return that is nearly optimal when averaged across all visited states, but this criterion turns out to be equivalent to maximizing average return—the discount factor ultimately plays no role. Ultimately, Kearns and Singh opt for finding a

near-optimal policy from the final state reached by the algorithm. In fact, we show that averaging discounted return is a meaningful criterion if it is the loss (relative to the optimal policy from each visited state) that is averaged.

While Kearns and Singh's notion of efficiency applies to a more general reinforcement-learning problem than does Fiechter's, it still includes an unnatural separation between learning and evaluation. Kakade [10] introduced a PAC performance metric that is more "online" in that it evaluates the behavior of the learning algorithm itself as opposed to a separate policy that it outputs. As in Kearns and Singh's definition, learning takes place over one long path through the MDP. At time t, the partial path $c_t = s_1, a_1, r_1, \ldots, s_t$ is used to determine a next action a_t. The algorithm itself can be viewed as a non-stationary policy. In our notation, this policy has expected value $V^{\mathcal{A}}(c_t)$, where $\mathcal{A}$ is the learning algorithm.

Definition 1. (See [10].) Let $c = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots)$ be a path generated by executing an algorithm $\mathcal{A}$ in an MDP M. For any fixed $\epsilon > 0$, the **sample complexity of exploration** (**sample complexity**, for short) of $\mathcal{A}$ with respect to c is the number of timesteps t such that the policy at time t, $\mathcal{A}_t$, is not ϵ-optimal from the current state, s_t at time t (formally, $V^{\mathcal{A}_t}(s_t) < V^*(s_t) - \epsilon$).

In other words, the sample complexity is the number of timesteps, over the course of any run, for which the learning algorithm $\mathcal{A}$ is not executing an ϵ-optimal policy from its current state. $\mathcal{A}$ is PAC in this setting if its sample complexity can be bounded by a number polynomial in the relevant quantities with high probability. Kakade showed that the R-max algorithm [3] satisfies this condition. We will use Kakade's [10] definition as the standard.

Definition 2. An algorithm $\mathcal{A}$ is said to be an **efficient PAC-MDP** (Probably Approximately Correct in Markov Decision Processes) algorithm if, for any ϵ and δ, the per-step computational complexity and the sample complexity of $\mathcal{A}$ are less than some polynomial in the relevant quantities $(|S|, |A|, 1/\epsilon, 1/\delta, 1/(1-\gamma))$, with probability at least $1-\delta$. For convenience, we may also say that $\mathcal{A}$ is **PAC-MDP**.

One thing to note is that we only restrict a PAC-MDP algorithm from behaving poorly (non-ϵ-optimally) on more than a small (polynomially) number of timesteps. We do not place any limitations on when the algorithm acts poorly. This is in contrast to the original PAC notion which is more "off-line" in that it requires the algorithm to make all its mistakes ahead of time before identifying a near-optimal policy.

This difference is necessary. In any given MDP it may take an arbitrarily long time to reach some section of the state space. Once that section is reached we expect any learning algorithm to make some mistakes. Thus, we can hope only to bound the number of mistakes, but can say nothing about when they happen. The first two performance metrics above were able to sidestep this issue somewhat. In Fiechter's framework, a reset action allows a more "offline" PAC-MDP definition. In the performance metric used by Kearns and Singh [11], a near-optimal policy is required only from a single state.

A second major difference between our notion of PAC-MDP and Valiant's original definition is that we do not require an agent to know when it has found a near-optimal policy, only that it executes one most of the time. In situations where we care only about the behavior of an algorithm, it does not make sense to require an agent to estimate its policy. In other situations, where there is a distinct separation between learning (exploring) and acting (exploiting), another performance metric, such as one of first two mentioned above, should be used. Note that requiring the algorithm to "know" when it has adequately learned a task may require the agent to explicitly estimate the value of its current policy. This may complicate the algorithm (for example, E^3 solves two MDP models instead of one).

Although sample complexity demands a tight integration between behavior and evaluation, the evaluation itself is still in terms of the near-optimality of expected values over future policies as opposed to the actual rewards the algorithm achieves while running. We introduce a new performance metric, *average loss*, defined in terms of the actual rewards received by the algorithm while learning. In the remainder of the section, we define average loss formally. In Section 5, we show that efficiency in the sample-complexity setting implies efficiency in the average-loss setting.

Definition 3. Suppose a learning algorithm is run for one trial of H steps in an MDP M. Let s_t be the state encountered on step t and let r_t be the tth reward received. Then, the **instantaneous loss** of the agent is $il(t) = V^*(s_t) - \sum_{i=t}^{H} \gamma^{i-t} r_i$, the difference between the optimal value function at state s_t and the actual discounted return of the agent from time t until the end of the trial. The quantity $l = \frac{1}{H}\sum_{t=1}^{H} il(t)$ is called the **average loss** over the sequence of states encountered.

In Definition 3, the quantity H should be sufficiently large, say $H \gg 1/(1-\gamma)$, because otherwise there is not enough information to evaluate the algorithm's performance. A learning algorithm is *PAC-MDP in the average loss setting* if for any ϵ and δ, we can choose a value H, polynomial in the relevant quantities $(1/\epsilon, 1/\delta, |S|, |A|, 1/(1-\gamma))$, such that the average loss of the agent (following the learning algorithm) on a trial of H steps is guaranteed to be less than ϵ with probability at least $1-\delta$.

It helps to visualize average loss in the following way. Suppose that an agent produces the following trajectory through an MDP:

$$s_1,\ a_1,\ r_1,\ s_2,\ a_2,\ r_2,\ \ldots,\ s_H,\ a_H,\ r_H.$$

The trajectory is made up of states, $s_t \in S$; actions, $a_t \in A$; and rewards, $r_t \in [0,1]$, for each timestep $t = 1, \ldots, H$. The instantaneous loss associated for each timestep is shown in the following table.

t	Trajectory starting at time t	Instantaneous loss: $il(t)$
1	$s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_H, a_H, r_H$	$V^*(s_1) - (r_1 + \gamma r_2 + \cdots + \gamma^{H-1} r_H)$
2	$s_2, a_2, r_2, \ldots, s_H, a_H, r_H$	$V^*(s_2) - (r_2 + \gamma r_3 + \cdots + \gamma^{H-2} r_H)$
.	.	.
.	.	.
.	.	.
H	s_H, a_H, r_H	$V^*(s_H) - r_H$

The average loss is then the average of the instantaneous losses (in the rightmost column above).

3. Formal description of Model-based Interval Estimation

Model-based Interval Estimation maintains the following variables for each state-action pair (s,a) of the MDP M.

- Action-value estimates $\tilde{Q}(s,a)$: These are used by the algorithm to select actions. They are rough approximations of $Q^*(s,a)$ and are computed by solving the internal MDP model used by MBIE. On timestep t, we denote $\tilde{Q}(s,a)$ by $\tilde{Q}_t(s,a)$. They are initialized optimistically; $\tilde{Q}_0(s,a) = 1/(1-\gamma)$.
- Reward estimates $\hat{R}(s,a)$: The average reward received for taking action a from state s. On timestep t, we denote $\hat{R}(s,a)$ by $\hat{R}_t(s,a)$.
- Transition estimates $\hat{T}(s,a)$: The maximum likelihood estimate of the true transition distribution $T(s,a)$. On timestep t, we denote $\hat{T}(s,a)$ by $\hat{T}_t(s,a)$.
- Occupancy counts $n(s,a)$: The number of times action a was taken from state s. On timestep t, we denote $n(s,a)$ by $n_t(s,a)$.
- Next-state counts $n(s,a,s')$ for each $s' \in S$: The number of times action a was taken from state s and resulted in next-state s'. On timestep t, we denote $n(s,a,s')$ by $n_t(s,a,s')$.

Besides these, there are several other quantities that MBIE uses.

- Inputs S, A, γ, ϵ, δ. These are provided as inputs to the algorithm before execution time.
- Model size limit m. The maximum number of experiences, per state-action pair (s,a), used to calculate the model parameters $\hat{R}(s,a)$ and $\hat{T}(s,a)$. After the first m experiences of (s,a), the algorithm ignores the data (immediate reward and next state) observed after any additional experiences of (s,a). The precise value of m can be chosen (as in Section 4.3) to ensure formal guarantees on the behavior of the algorithm or it can be given as input to the algorithm. In the former case, m will depend on the inputs to the algorithm and especially on ϵ and δ. In the latter case, the value can be chosen using domain knowledge.

When we discuss the behavior of the algorithm at some fixed and specified timestep t, we will often omit the subscript t in the above quantities.

On each timestep, the agent executing MBIE chooses an action greedily with respect to $\tilde{Q}(s,a)$. That is, if s_t is the tth state reached by the agent, then $a_t = \text{argmax}\{\tilde{Q}_t(s,a)\}$ is the tth action chosen. If there is a tie, then the algorithm may break the tie arbitrarily.

It is important to note that for each state-action pair (s,a), MBIE uses *only the first m experiences* of (s,a).[3] This means that if on timestep t, action a has been taken from state s more than m times, then the resulting immediate reward and next-state for those experiences are ignored. They are not used to update $\hat{R}(s,a)$ or $\hat{T}(s,a)$. In addition, $n(s,a)$ can be at most m (additional experiences of (s,a) after the mth one do not increase $n(s,a)$). The same is true of $n(s,a,s')$.

We now consider two different but similar algorithms, MBIE and MBIE-EB (short for *Model Based Interval Estimation with Exploration Bonus*). They differ only in the way $\tilde{Q}$ is computed. MBIE builds the upper tail of a confidence interval on the optimal value function of M by simultaneously considering a confidence interval on the entire space of MDPs. The approach taken by MBIE is closely related to that [24]. MBIE-EB uses a simpler form of the confidence intervals it maintains and is closely related to the confidence intervals computed by the Action Elimination algorithm [5]. It can be viewed as directly computing the upper tail of a confidence interval on the optimal value function of the MDP M. It is an open question which form of MBIE is better in practice. However, we suspect that the simpler version, MBIE-EB, will learn slightly more slowly but use less computation.

[3] This property of the algorithm is mainly enforced for our analysis. As it is hard to track the value of a changing, non-stationary policy, our analysis requires only a small (polynomial) number of changes to the agent's policy. Since any update to MBIE's model causes a new policy to be computed, we must place a restriction on the number of model updates. This also has the additional benefit of limiting the computational requirements of the algorithm. However, in experiments we have found that setting m to be very large or infinite actually improves the performance of the algorithm when computation is ignored.

3.1. MBIE's model

The action-value estimates $\tilde{Q}(s,a)$ are computed by solving an MDP model. In this section, we describe the model used by MBIE and a method for computing $\tilde{Q}$ from the model. This completes the specification of the MBIE algorithm.

We first provide an intuition behind the model. On timestep t, the algorithm makes use of two confidence intervals for each state-action pair (s,a), one on the mean reward $R(s,a)$ and another on the transition probability distribution $T(s,a)$. These two confidence intervals are then combined into a single confidence interval of MDPs on the underlying MDP M. The model used by MBIE is the MDP within this confidence interval whose optimal policy has the highest value from the current state of the agent (s_t).

3.1.1. The reward confidence interval

Suppose on timestep t, for state-action pair (s,a), the agent has observed the following $n(s,a)$ immediate rewards for taking action a from state s: $r[1], r[2], \ldots, r[n(s,a)]$. Then, the empirical mean reward is

$$\hat{R}(s,a) := \frac{1}{n(s,a)} \sum_{i=1}^{n(s,a)} r[i]. \tag{1}$$

The confidence interval for $R(s,a)$ is $CI(R) := (\hat{R}(s,a) - \epsilon^R_{n(s,a)}, \hat{R}(s,a) + \epsilon^R_{n(s,a)})$, where

$$\epsilon^R_{n(s,a)} := \sqrt{\frac{\ln(2/\delta_R)}{2n(s,a)}}. \tag{2}$$

We refer to $CI(R)$ as the *reward confidence interval* because of the following property. Assuming that the rewards $r[1], r[2], \ldots, r[n(s,a)]$ are independently drawn samples from $\mathcal{R}(s,a)$, we know from the Hoeffding bound that with probability at least $1-\delta_R$, $R(s,a) \in CI(R)$ will hold.

3.1.2. The transition confidence interval

For a fixed state-action pair (s,a) and timestep t, recall that $T(s,a)$ is the true transition probability distribution for (s,a). It can be viewed as an $|S|$-dimension vector where each component is between 0 and 1. On a fixed timestep t, the *empirical transition vector* is the vector $\hat{T}(s,a)$ with components

$$\hat{T}(s'|s,a) := \frac{n(s,a,s')}{n(s,a)}. \tag{3}$$

The right-hand side of Eq. (3) is the proportion of the number of times, among the first m experiences of (s,a) up to timestep t, that the resulting next-state was s'. The confidence interval for $T(s,a)$ is

$$CI(T) := \left\{ \tilde{T}(s,a) \in \mathcal{P}_S \;\middle|\; \left\| \tilde{T}(s,a) - \hat{T}(s,a) \right\|_1 \leqslant \epsilon^T_{n(s,a)} \right\} \tag{4}$$

where $\mathcal{P}_S$ is the set of all $|S|$-dimensional probability vectors (each component is between 0 and 1 and the sum of all components is 1) and

$$\epsilon^T_{n(s,a)} = \sqrt{\frac{2[\ln(2^{|S|}-2) - \ln(\delta_T)]}{m}}. \tag{5}$$

It is important to keep in mind that $CI(T)$ is a set (and, in fact, a confidence interval) of probability distributions over the finite state space S.

Let $s[1], s[2], \ldots, s[n(s,a)]$ be the $n(s,a)$ next-states observed by the agent after taking action a from state s and used to compute $\hat{T}(s,a)$. We refer to $CI(T)$ as the *transition confidence interval* because of the following property. Assuming these $n(s,a)$ next-states are independently drawn samples from $T(s,a)$, we have that with probability at least $1-\delta_T$, $T(s,a) \in CI(T)$ will hold [23].

Q(s,a) \in \[minR(s,a) + min(…..); \]

3.1.3. Combining the confidence intervals

On every timestep t, the main computation of the MBIE is to solve the following set of $|S||A|$ equations for $\tilde{Q}(\cdot,\cdot)$.

$$\tilde{Q}(s,a) = \max_{\tilde{R}(s,a) \in CI(R)} \tilde{R}(s,a) + \max_{\tilde{T}(s,a) \in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a'). \tag{6}$$

Note that this expression effectively combines the uncertainty in the rewards and transitions to provide the MDP model used by MBIE. If a state-action pair (s,a) has never been experienced by the agent, then it is not clear what $CI(R)$ and $CI(T)$ should be. In this case we simply define $\tilde{Q}(s,a) = 1/(1-\gamma)$ for those (s,a) for which $n(s,a) = 0$, a process referred to as *optimistic initialization*.

It is not immediately clear how to quickly solve Eq. (6) for each (s,a) simultaneously. First, note that the reward term (the first part of the right-hand side) is maximized easily. Using this we can simplify Eq. (6) to $\tilde{Q}(s,a) = (\hat{R}(s,a) + \epsilon^R_{n(s,a)}) +$

$\max_{\tilde{T}(s,a)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a')$. The set of equations now looks a lot like the Bellman equations for solving a single MDP. However, the additional maximization over transition probability distributions complicates the situation and rules out a straight-forward reduction to MDP solving.

3.1.4. Value iteration with confidence intervals

In this section, we argue that Eq. (6) can be solved using value iteration. This problem is strongly related to the problem considered by Givan et al. [8] that they call finding an "optimistically optimal policy." The main difference is that in their problem, confidence intervals (on the transition distributions) are specified by a bound on the L_∞ rather than L_1 norms. The attainment of our sample-complexity bounds require the use of L_1 norms. The problem of finding a policy that is no worse than the optimal policy for any MDP whose transition function falls in the given confidence interval is interesting but different than our problem and has been well studied [8,14].

Suppose we start with an arbitrary value function, $Q : S \times A \to \mathbb{R}$. Then, we can obtain a new value function, Q', by using Eq. (6) as an assignment statement:

$$Q'(s,a) := \max_{\tilde{R}(s,a)\in CI(R)} \tilde{R}(s,a) + \max_{\tilde{T}(s,a)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s,a) \max_{a'} Q(s',a'). \tag{7}$$

This procedure can be repeated indefinitely to produce a sequence of value functions, which we claim converges to the unique optimal solution to $\tilde{Q}$ of Eq. (6). This follows from the fact that Eq. (6) leads to a contraction mapping.

Proposition 1. *Let $Q(s,a)$ and $W(s,a)$ be value functions and $Q'(s,a)$ and $W'(s,a)$ be the result of applying Eq. (7) to Q and W, respectively. The update results in a contraction mapping in max-norm, that is,* $\max_{(s,a)} |Q'(s,a) - W'(s,a)| \leqslant \gamma \max_{(s,a)} |Q(s,a) - W(s,a)|$.

Proof. The argument is nearly identical to the standard convergence proof of value iteration [15], noting that the maximization over the compact set of probability distributions does not interfere with the contraction property. Let $(s^*,a^*) = \operatorname{argmax}_{(s,a)} |Q'(s,a) - W'(s,a)|$, and let $CI(T)$ denote the transition confidence interval (see Eq. (4)) for the state-action pair (s^*,a^*). Let $V(s) := \max_a Q(s,a)$ and $X(s) := \max_a W(s,a)$ for all states s. Let

$$T_1 := \operatorname*{argmax}_{\tilde{T}(s^*,a^*)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s^*,a^*)V(s') \quad \text{and} \quad T_2 := \operatorname*{argmax}_{\tilde{T}(s^*,a^*)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s^*,a^*)X(s').$$

Without loss of generality, assume that $Q'(s^*,a^*) \geqslant W'(s^*,a^*)$. Then,

$$\begin{aligned}
\max_{s,a} \big|Q'(s,a) - W'(s,a)\big| &= Q'(s^*,a^*) - W'(s^*,a^*) \\
&= \max_{\tilde{T}(s^*,a^*)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s^*,a^*)V(s') - \max_{\bar{T}(s^*,a^*)\in CI(T)} \gamma \sum_{s'} \bar{T}(s'|s^*,a^*)X(s') \\
&= \gamma \sum_{s'} T_1(s'|s^*,a^*)V(s') - \gamma \sum_{s'} T_2(s'|s^*,a^*)X(s') \\
&\leqslant \gamma \sum_{s'} T_1(s'|s^*,a^*)V(s') - \gamma \sum_{s'} T_1(s'|s^*,a^*)X(s') \\
&\leqslant \gamma \max_s \big|V(s) - X(s)\big| \leqslant \gamma \max_{(s,a)} \big|Q(s,a) - W(s,a)\big|.
\end{aligned}$$

The first inequality results from the definition of T_2. Specifically, T_2 is the probability distribution in $CI(T)$ that maximizes the expected value of the next state under $X(\cdot)$, so it is greater than the expected value with respect to the distribution T_1. The rest of the steps follow from basic algebraic manipulation. □

3.1.5. Solving the CI constraint

To implement value iteration based on Eq. (6), we need to solve the following computational problem. Maximize

$$M_V = \sum_{s'} \tilde{T}(s'|s,a)V(s'),$$

Does this really need to happen every iteration? Cant we store the ordering and 'detect' if something changes?

over all probability distributions, $\tilde{T}(\cdot|s,a)$, subject to the constraint $\|\tilde{T}(\cdot|s,a) - \hat{T}(\cdot|s,a)\|_1 \leqslant \varepsilon$, where $\varepsilon = \epsilon^T_{n(s,a)}$ as in Eq. (5).

The following procedure can be used to find the maximum value. Let $\tilde{T}(\cdot|s,a) = \hat{T}(\cdot|s,a)$ to start. Let $s^* = \operatorname{argmax}_s V(s)$ and increment $\tilde{T}(s^*|s,a)$ by $\varepsilon/2$. At this point, $\tilde{T}(\cdot|s,a)$ is not a probability distribution, since it sums to $1 + \varepsilon/2$; however, its L_1 distance to $\hat{T}(\cdot|s,a)$ is $\varepsilon/2$. We need to remove exactly $\varepsilon/2$ weight from $\tilde{T}(\cdot|s,a)$. The weight is removed iteratively, taking the maximum amount possible from the state $s^- = \operatorname{argmin}_{s'|\tilde{T}(s'|s,a)>0} V(s')$, since this decreases M_V the least.

Proposition 2. *The procedure just described maximizes M_V.*

Proof. First, the weight on s^* is at most $\hat{T}(s^*|s,a) + \varepsilon/2$. If it is larger, $\tilde{T}(\cdot|s,a)$ violates the L_1 constraint. Let $\tilde{T}(s'|s,a) - \hat{T}(s'|s,a)$ be called the *residual* of state s'. If $\|\tilde{T}(\cdot|s,a) - \hat{T}(\cdot|s,a)\|_1 = \varepsilon$, then the sum of the positive residuals is $\varepsilon/2$ and the sum of the negative residuals is $-\varepsilon/2$. For two states s_1 and s_2, if $V(s_1) > V(s_2)$, then M_V is increased by moving positive residual from s_2 to s_1 or negative residual from s_1 to s_2. Therefore, putting all positive residual on s^* and moving all the negative residual toward states with smallest $V(s)$ values maximizes M_V among all distributions $\tilde{T}(\cdot|s,a)$ with the given L_1 constraint. □

3.2. MBIE-EB's model

MBIE-EB simply solves the following set of equations to compute its action-value estimates, $\tilde{Q}(\cdot,\cdot)$:

$$\tilde{Q}(s,a) = \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a') + \frac{\beta}{\sqrt{n(s,a)}}, \tag{8}$$

where $\hat{R}$ and $\hat{T}$ form the empirical model based on the first m experiences for each state-action pair as described above and β is a constant provided as input to the algorithm. Eq. (8) can be solved efficiently using any technique for solving an MDP.

4. Analysis

In this section we provide a formal analysis of both versions of the MBIE algorithm. Throughout this section we measure complexity assuming individual numbers require unit storage and can be manipulated arithmetically in unit time. Removing this assumption increases space and computational complexities by logarithmic factors.

4.1. Computation complexity of MBIE

The computational complexity of MBIE depends on the implementation. Our complexity bounds reflect one possible implementation of the algorithm. In this section we first discuss the worst case per-timestep computational complexity of the algorithm. Then we discuss the worst case total computational complexity. The per-timestep complexity is interesting from a learning viewpoint, where the longest time between two actions may be more important than the total time taken over all actions.

Consider some timestep t during execution of MBIE. Suppose that s is the tth state reached, a is the tth action chosen, r is the resulting immediate reward, and s' is the next-state.

At this point, the first computation of the algorithm is to update its internal model. That is, it updates the variables $\hat{R}(s,a)$, $\hat{T}(s,a)$, $n(s,a)$, and $n(s,a,s')$, which can be done in constant time. The next computation is to recalculate $\tilde{Q}(s,a)$ for each (s,a) using value iteration as described in Section 3.1.3. Each iteration of value iteration involves a loop through each state-action pair of the MDP. Within this loop, the algorithm performs the optimization described in Section 3.1.5. This optimization sorts the observed reachable next-states by their estimated value, which takes $\Theta(|S|\log_2(|S|)) = \Theta(|S|\ln(|S|))$ computation time. It also requires knowledge of the state s^* (see Section 3.1.5) whose current value estimate is maximum. To obtain this quickly, our implementation of MBIE includes a global priority queue[4] of size $|S||A|$ that holds all the action-value estimates for each state. Finally, the algorithm must choose an action. To facilitate this choice, we also maintain a local priority queue for each state s that stores the values, $\tilde{Q}(s,a)$, for each action a.

To summarize, if $\mathcal{N}$ is the maximum number of iterations of value iteration used to solve Eq. (6) during execution of MBIE on any timestep, then the worst case per-timestep computational complexity of MBIE is $O(|S||A|\mathcal{N}(|S|\ln(|S|) + |S| + \ln(|S||A|) + \ln(|A|))) = O(|S||A|\mathcal{N}(|S|\ln(|S|) + \ln(|S||A|)))$. The $|S||A|\mathcal{N}$ factor is the number of updates performed during value iteration. The $|S|\ln(|S|)$ term is for sorting the next-states by value. The $|S| + \ln(|S||A|)$ term follows from the cost of computing the new action-value estimate for a single state-action pair and updating the global priority queue. The $\ln(|A|)$ term follows from the fact that the new action-value estimate needs to be stored in a local priority queue for the current state.

For simplicity in our analysis and description of the algorithm, we have assumed that MBIE solves its model (specified by Eq. (6)) exactly. However, it is easy to modify the analysis to show that we only need value iteration to produce a $C\epsilon$-optimal policy (rather than an optimal one), for some constant C. Thus, we can easily bound $\mathcal{N}$ by a polynomial[5] in the input parameters. In practice, we have found that it works well to stop once successive iterations do not result in a change to any action-value estimate $\tilde{Q}(s,a)$ that is greater than some small threshold. The space complexity of MBIE is $O(m|S||A|)$, which can be achieved by maintaining the counts $n(s,a,s')$ only for those s' that have been reached by the agent during the first m experiences of (s,a).

[4] In [20] an approximation that removes the necessity of maintaining this priority queue was introduced. The approximation is simply to use the value $1/(1-\gamma)$ as a bound on the value of any state.

[5] Specifically, it is well known that $\mathcal{N} = O(\ln(\frac{1}{\epsilon(1-\gamma)})/(1-\gamma))$ is sufficient.

4.1.1. Total computation time

We have shown that the per-step computation time of our implementation of MBIE is $O(|S||A|\mathcal{N}(|S|\ln(|S|)+\ln(|S||A|)))$, where $\mathcal{N}$ is an upper bound on the number of iterations used by the version of value iteration described in Section 3.1.4. Note that on each timestep, MBIE performs more than a constant amount of computation only if it updates its internal model. If no model update is required, than MBIE simply needs to choose an action by solving $\operatorname{argmax}_{a\in A}\tilde{Q}(s,a)$, where s is the current state. This can be done in constant time by using a priority queue as discussed above. Since MBIE can update its model at most $|S||A|m$ times, the total computation complexity is

$$O\big(B+|S|^2|A|^2m\mathcal{N}\big(|S|\ln\big(|S|\big)+\ln\big(|S||A|\big)\big)\big) \tag{9}$$

where B is the number of timesteps for which MBIE is executed.

Prioritized sweeping?
Do we need to update every time step?
Can we update only if something changes significantly?
Do we increment counts if relative ordering remains the same?

4.2. Computation complexity of MBIE-EB

As we have done for MBIE, we first consider the worst case per-timestep computational complexity of the algorithm. In fact, the only difference, in terms of computation time, between the two algorithms is that MBIE-EB does not need to perform the optimization step of Section 3.1.5. Instead it can perform traditional value iteration on its model. This incurs a per-step computational cost of $O(|S||A|\mathcal{N}(|S|+\ln(|A|)))$, where $\mathcal{N}$ is the maximum number of iterations of value iteration used to solve Eq. (8) during execution of MBIE-EB on any timestep. After each action-value update, a cost of $\ln(|A|)$ is incurred for storing the new action-value estimate in a local priority queue for the current state. Like MBIE, the space complexity of MBIE is $O(m|S||A|)$.

4.2.1. Total computation time

We have shown that the per-step computation time of our implementation of MBIE-EB is $O(|S||A|\mathcal{N}(|S|+\ln(|A|)))$. This leads to a total computation complexity of

$$O\big(B+|S|^2|A|^2m\mathcal{N}\big(|S|+\ln\big(|A|\big)\big)\big) \tag{10}$$

where B is the number of timesteps for which MBIE-EB is executed.

4.3. Sample complexity analysis

In this section, we study the sample complexity of exploration of MBIE. Our main result is summarized by the following theorem.

Theorem 1. *Suppose that ϵ and δ are two real numbers between 0 and 1 and $M=\langle S,A,T,\mathcal{R},\gamma\rangle$ is any MDP. There exists inputs $\delta_R=\delta_T=\delta/(2|S||A|m)$ and $m=m(\frac{1}{\epsilon},\frac{1}{\delta})$, satisfying $m(\frac{1}{\epsilon},\frac{1}{\delta})=O(\frac{|S|}{\epsilon^2(1-\gamma)^4}+\frac{1}{\epsilon^2(1-\gamma)^4}\ln\frac{|S||A|}{\epsilon(1-\gamma)\delta})$, such that if MBIE is executed on M, then the following holds. Let $\mathcal{A}_t$ denote MBIE's policy at time t and s_t denote the state at time t. With probability at least $1-\delta$, $V_M^{\mathcal{A}_t}(s_t)\geqslant V_M^*(s_t)-\epsilon$ is true for all but $O(\frac{|S||A|}{\epsilon^3(1-\gamma)^6}(|S|+\ln\frac{|S||A|}{\epsilon(1-\gamma)\delta})\ln\frac{1}{\delta}\ln\frac{1}{\epsilon(1-\gamma)})$ timesteps t.*

At the beginning of a run, every state-action (s,a) pair is said to be *unknown*. At any step of the algorithm, the set of *known* state-action pairs K is defined to be those (s,a) experienced at least m times [11]. For large enough m, any $(s,a)\in K$ will be accurately modeled.

An overview of the sample-complexity analysis is as follows. At each timestep, MBIE follows the optimal policy of its model $\tilde{M}$. Lemmas 2, 3, and 5 show that the value of MBIE's policy in its model is very close to its true value as long as the probability of reaching an unknown state-action pair is low. By Lemma 6, the estimated value of its policy is at least as large, with high probability, as the true optimal value function. Thus, MBIE chooses its actions based on a policy that is either nearly optimal or one with a high probability of encountering an unknown (s,a). However, the number of times a given (s,a) can be experienced before it becomes known is shown to be no more than polynomial in the relevant quantities. Therefore, the agent will act nearly optimally on all but a bounded number of timesteps—it has polynomial sample complexity.

Our entire analysis is grounded on the idea of using samples, in the form of immediate rewards and next-states, to estimate the reward and transition probability distributions for each state-action pair. The main analytical tools we use are large deviation bounds such as the Hoeffding bound (see, for instance, [12]). The Hoeffding bound allows us to quantify a number of samples sufficient to guarantee, with high probability, an accurate estimate of an unknown quantity (for instance, the transition probability to some next-state). However, its use requires *independent* samples. It may appear at first that the immediate reward and next-state observed after taking a fixed action a from a fixed state s is independent of all past immediate rewards and next-states observed. Indeed, due to the Markov property, the immediate reward and next-state are guaranteed to be independent of the entire history of the agent given the current state. However, there is a subtle way in which the samples may not be independent. In Appendix A, we discuss this issue in detail and show that our use of large deviation bounds still hold.

4.3.1. Simulation properties for discounted MDPs

In this section we investigate the notion of using one MDP as a model or simulator of another MDP. Specifically, suppose that we have two MDPs, M_1 and M_2, with the same state and action space and discount factor. We ask how similar must the transitions and rewards of M_1 and M_2 be in order to guarantee that difference between the value of a fixed policy π in M_1 and its value in M_2 is no larger than some specified threshold ϵ. Although we are not able to answer the question completely, we do provide a sufficient condition (Lemma 2) that uses L_1 distance to measure the difference between the two transition distributions. Finally, we end with a result (Lemma 3) that measures the difference between a policy's value in the two MDPs when they have equal transitions and rewards most of the time but are otherwise allowed arbitrarily different transitions and rewards.

The following lemma helps develop Lemma 2, a slight improvement over the "Simulation Lemma" of [11] for the discounted case. We defer its proof to Appendix B. In the next two lemmas we allow for the possibility of rewards greater than 1 (but still bounded) because they may be of interest outside of the present work. However, for the rest of the paper, we continue to assume, unless otherwise specified, that all rewards fall in the interval $[0, 1]$.

Lemma 1. *Let $M_1 = \langle S, A, T_1, R_1, \gamma\rangle$ and $M_2 = \langle S, A, T_2, R_2, \gamma\rangle$ be two MDPs with non-negative rewards bounded by $R_{\max}$. If $|R_1(s,a) - R_2(s,a)| \leqslant \alpha$ and $\|T_1(s,a,\cdot) - T_2(s,a,\cdot)\|_1 \leqslant \beta$ for all states s and actions a, then the following condition holds for all states s, actions a, and stationary, deterministic policies π:*

$$\left|Q_1^\pi(s,a) - Q_2^\pi(s,a)\right| \leqslant \frac{\alpha + \gamma R_{\max}\beta}{(1-\gamma)^2}.$$

Algorithms like MBIE act according to an internal model. The following lemma shows that two MDPs with similar transition and reward functions have similar value functions. Thus, an agent need only ensure accuracy in the transitions and rewards of its model to guarantee near-optimal behavior.

Lemma 2. *Let $M_1 = \langle S, A, T_1, R_1, \gamma\rangle$ and $M_2 = \langle S, A, T_2, R_2, \gamma\rangle$ be two MDPs with non-negative rewards bounded by $R_{\max}$, which we assume is at least 1. Suppose that $|R_1(s,a) - R_2(s,a)| \leqslant \alpha$ and $\|T_1(s,a,\cdot) - T_2(s,a,\cdot)\|_1 \leqslant \beta$ for all states s and actions a. There exists a constant C, such that for any $0 < \epsilon \leqslant R_{\max}/(1-\gamma)$ and stationary policy π, if $\alpha = \beta = C(\frac{\epsilon(1-\gamma)^2}{R_{\max}})$, then*

$$\left|Q_1^\pi(s,a) - Q_2^\pi(s,a)\right| \leqslant \epsilon. \tag{11}$$

Proof. By Lemma 1, we have that $|Q_1^\pi(s,a) - Q_2^\pi(s,a)| \leqslant \frac{\alpha(1+\gamma R_{\max})}{(1-\gamma)^2}$. Thus, it is sufficient to guarantee that $\alpha \leqslant \frac{\epsilon(1-\gamma)^2}{1+\gamma R_{\max}}$. We choose $C = 1/2$ and by our assumption that $R_{\max} \geqslant 1$ we have that $\alpha = \frac{\epsilon(1-\gamma)^2}{2R_{\max}} \leqslant \frac{\epsilon(1-\gamma)^2}{1+\gamma R_{\max}}$. □

The following lemma relates the difference between a policy's value function in two different MDPs, when the transition and reward dynamics for those MDPs are identical on some of the state-action pairs (those in the set K), and arbitrarily different on the other state-action pairs. When the difference between the value of the same policy in these two different MDPs is large, the probability of reaching a state that distinguishes the two MDPs is also large.

Lemma 3 *(Generalized induced inequality). Let M be an MDP, K a set of state-action pairs, M' an MDP equal to M on K (identical transition and reward functions), π a policy, and H some positive integer. Let A_M be the event that a state-action pair not in K is encountered in a trial generated by starting from state s_1 and following π for H steps in M. Then,*

$$V_M^\pi(s_1, H) \geqslant V_{M'}^\pi(s_1, H) - \big(1/(1-\gamma)\big)\Pr(A_M).$$

Proof. For some fixed *partial path* $p_t = s_1, a_1, r_1, \ldots, s_t, a_t, r_t$, let $P_{t,M}(p_t)$ be the probability p_t resulted from execution of policy π in M starting from state s_1. Let K_t be the set of all paths p_t such that every state-action pair (s_i, a_i) with $1 \leqslant i \leqslant t$ appearing in p_t is "known" (in K). Let $r_M(t)$ be the reward received by the agent at time t, and $r_M(p_t, t)$ the reward at time t given that p_t was the partial path generated. Now, we have the following:

$$\begin{aligned}
E\big[r_{M'}(t)\big] - E\big[r_M(t)\big] &= \sum_{p_t \in K_t} \big(P_{t,M'}(p_t) r_{M'}(p_t,t) - P_{t,M}(p_t) r_M(p_t,t)\big) + \sum_{p_t \notin K_t} \big(P_{t,M'}(p_t) r_{M'}(p_t,t) - P_{t,M}(p_t) r_M(p_t,t)\big) \\
&= \sum_{p_t \notin K_t} \big(P_{t,M'}(p_t) r_{M'}(p_t,t) - P_{t,M}(p_t) r_M(p_t,t)\big) \\
&\leqslant \sum_{p_t \notin K_t} P_{t,M'}(p_t) r_{M'}(p_t,t) \leqslant \Pr(A_M).
\end{aligned}$$

The first step in the above derivation involved separating the possible paths in which the agent encounters an unknown state-action from those in which only known state-action pairs are reached. We can then eliminate the first term, because M

and M' behave identically on known state-action pairs. The last inequality makes use of the fact that all rewards are at most 1. The result then follows from the fact that $V^{\pi}_{M'}(s_1, H) - V^{\pi}_{M}(s_1, H) = \sum_{t=0}^{H-1} \gamma^t (E[r_{M'}(t)] - E[r_M(t)])$. □

The following well-known result allows us to truncate the infinite-horizon value function for a policy to a finite-horizon one.

Lemma 4. *If $H \geqslant \frac{1}{1-\gamma} \ln \frac{1}{\epsilon(1-\gamma)}$ then $|V^{\pi}(s, H) - V^{\pi}(s)| \leqslant \epsilon$ for all policies π and states s.*

Proof. See Lemma 2 of [11]. □

4.3.2. Sample complexity of MBIE

As the MBIE agent gathers experience, it is continuously updating and solving its model of the world according to Eq. (6). Let *CI* be any confidence interval computed by MBIE. We say that *CI* is *admissible* if it contains the mean of the distribution that produced the samples for which *CI* was computed from. For our following analysis, we require that all confidence intervals—reward and transition—be admissible for all state-action pairs over every time-step, with high probability. Note that MBIE will compute at most $|S||A|m$ confidence intervals (m confidence intervals per state-action pair). Thus, to ensure admissible confidence intervals with probability at least $1 - \delta$, it is sufficient to set $\delta_T = \delta_R = \delta/(2|S||A|m)$ and apply the union bound.

The next lemma quantifies the amount of experience, for each state-action pair, required by MBIE to accurately model the dynamics of the environment.

Lemma 5. *Suppose that $\delta_T = \delta_R = \delta/(2|S||A|m)$ and that all confidence intervals computed by MBIE are admissible. Then, for any $\tau > 0$, there exists an $m = O(\frac{|S|}{\tau^2} + \frac{1}{\tau^2} \ln \frac{|S||A|}{\tau\delta})$ such that $|\tilde{R}(s,a) - R(s,a)| \leqslant \tau$ and $\|\tilde{T}(s,a,\cdot) - T(s,a,\cdot)\|_1 \leqslant \tau$ holds for all state-action pairs (s,a) that have been experienced at least m times.*

Proof. Consider a fixed state-action pair (s,a) and some fixed timestep t during execution of MBIE. From Eq. (2), the size of the reward confidence interval associated with (s,a) is $2\sqrt{\frac{\ln(2/\delta_R)}{2n(s,a)}}$. From Eq. (5), the size of the corresponding transition confidence interval is $2\sqrt{\frac{2[\ln(2^{|S|}-2) - \ln(\delta_T)]}{n(s,a)}}$. Once (s,a) has been experienced m times by the agent, these intervals become fixed, $n(s,a) = m$, and the two expressions above become $2\sqrt{\frac{\ln(2/\delta_R)}{2m}}$ and $2\sqrt{\frac{2[\ln(2^{|S|}-2) - \ln(\delta_T)]}{m}}$. Using the assumption that both confidence intervals are admissible, we have that

$$m \geqslant \max\left\{ \frac{8[\ln(2^{|S|} - 2) - \ln(\delta_T)]}{\tau^2}, \frac{2\ln(2/\delta_R)}{\tau^2} \right\} \tag{12}$$

is sufficient to guarantee that $|\tilde{R}(s,a) - R(s,a)| \leqslant \tau$ and $\|\tilde{T}(s,a,\cdot) - T(s,a,\cdot)\|_1 \leqslant \tau$ when $n(s,a) = m$. By substitution of $\delta/(2|S||A|m)$ for δ_R and δ_T, the right-hand side of Eq. (12), it can be rewritten as $\max\{\frac{8[\ln(2^{|S|}-2) + \ln(2|S||A|m/\delta)]}{\tau^2}, \frac{2\ln(4|S||A|m/\delta)}{\tau^2}\}$. It is a well-known fact that for any constant $C > 0$, $n \geqslant 2C\ln(C)$ implies $n \geqslant C\ln(n)$. Using this, it is clear that m can be chosen large enough so that Eq. (12) holds, but small enough so that $m = O(\frac{|S|}{\tau^2} + \frac{1}{\tau^2} \ln \frac{|S||A|}{\epsilon\delta})$. □

The MBIE algorithm exhibits "optimism in the face of uncertainty." This notion is captured formally by the following lemma. Specifically, we show that the expected return of acting in the MBIE agent's model is at least as large as the expected return of acting in the underlying environment, with high probability.

Lemma 6. *Suppose that all confidence intervals computed by MBIE are admissible. Then, for any state s and action a, the condition $\tilde{Q}(s,a) \geqslant Q^*(s,a)$ is satisfied during any iteration of MBIE.*

Proof. At each step of the learning problem, MBIE solves the MDP $\tilde{M}$. We prove the claim by induction on the number of steps of value iteration.[6] For the base case, assume that the Q values are initialized to $1/(1-\gamma) \geqslant V^*(s)$, for all s. Now, for the induction, suppose that the claim holds for the current value function $\tilde{Q}(s,a)$.

MBIE computes two confidence intervals. $CI(R)$ is an interval of real numbers of the form $(\hat{R}(s,a) - \epsilon^R_{n(s,a)}, \hat{R}(s,a) + \epsilon^R_{n(s,a)})$. $CI(T)$ is the set of probability distributions $T'(\cdot|s,a)$ of the form $\|\hat{T}(\cdot|s,a) - T'(\cdot|s,a)\|_1 \leqslant \epsilon^T_{n(s,a)}$. By assumption, we have that $R(s,a) \in CI(R)$ and $T(s,a) \in CI(T)$.

The term $\tilde{Q}(s',a')$ on the right-hand side of Eq. (6) is the result of the previous iteration and is used to compute the new Q-value $\tilde{Q}(s,a)$ on the left-hand side of the equation. By our confidence-interval assumption, we have $\tilde{R}(s,a) \geqslant R(s,a)$ and

[6] We assume here that value iteration is halted after a finite number of iterations.

$$\max_{\tilde{T}(\cdot|s,a)\in CI(T)} \gamma \sum_{s'} \tilde{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a') \geqslant \gamma \sum_{s'} T(s'|s,a) \max_{a'} \tilde{Q}(s',a')$$
$$\geqslant \gamma \sum_{s'} T(s'|s,a) \max_{a'} Q^*(s',a').$$

The first step follows from the assumption that $T(\cdot|s,a) \in CI(T)$ and the second from the induction assumption. □

We are now ready to prove Theorem 1.

Proof of Theorem 1. Let ϵ_1 be an arbitrary positive real number, whose precise value we will specify later. We assume that all confidence intervals computed by MBIE are admissible, an assumption that holds with probability at least $1-\delta/2$, by the union bound and our choice of δ_R and δ_T. Using Lemma 4, let $H = O(\frac{1}{1-\gamma} \ln \frac{1}{\epsilon_1(1-\gamma)})$ be a positive integer large enough so that for all MDPs M' with discount factor γ, policies π, and states s, $|V^{\pi}_{M'}(s,H) - V^{\pi}_{M'}(s)| \leqslant \epsilon_1$.

Consider some fixed timestep t. Let s_t be the current state of the agent. Define K to be the set of all state-action pairs (s,a) that have been experienced at least m times ($n(s,a) \geqslant m$); we will provide the value of m shortly. We call K the set of *known state-action pairs*. Recall that the MBIE agent (denoted by policy $\mathcal{A}_t$) chooses its next action by following an optimal policy $\tilde{\pi}$ of MBIE's internal model $\tilde{M}$ at time t. Let M' be the MDP that is equal to M on K (meaning equal reward and transition distributions for $(s,a) \in K$) and equal to $\tilde{M}$ on $S \times A - K$. Using Lemmas 2, 5 (with $\tau = C\epsilon_1(1-\gamma)^2$, where C is the constant specified in Lemma 2), and our assumption of admissible confidence intervals, it follows that we can choose m with $m = O(\frac{|S|}{\epsilon_1^2(1-\gamma)^4} + \frac{1}{\epsilon_1^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon_1(1-\gamma)\delta})$ so that

$$\left|V^{\tilde{\pi}}_{M'}(s) - V^{\tilde{\pi}}_{\tilde{M}}(s)\right| \leqslant \epsilon_1 \tag{13}$$

holds for all s. Let A_M be the event that $\tilde{\pi}$ "escapes" from K in H steps. Specifically, A_M is the event that some $(s,a) \notin K$ is experienced after following $\tilde{\pi}$ from state s_t for H-steps. We claim that for all states s:

$$V^{\mathcal{A}_t}_M(s,H) \geqslant V^{\tilde{\pi}}_{M'}(s,H) - \Pr(A_M)/(1-\gamma). \tag{14}$$

This follows from Lemma 3 and the fact that the policies $\mathcal{A}_t$ and $\tilde{\pi}$ behave identically unless A_M occurs.

We now consider two mutually exclusive cases. First, suppose that $\Pr(A_M) \geqslant \epsilon_1(1-\gamma)$, meaning that an agent following $\mathcal{A}_t$ will encounter an unknown (s,a) in H steps with probability at least $\epsilon_1(1-\gamma)$. Using the Hoeffding bound,[7] after $O(\frac{m|S||A|H}{\epsilon_1(1-\gamma)} \ln \frac{1}{\delta})$ timesteps t where $\Pr(A_M) \geqslant \epsilon_1(1-\gamma)$ is satisfied, all (s,a) will become known, with probability at least $1-\delta/2$.

Now, suppose that $\Pr(A_M) < \epsilon_1(1-\gamma)$. Then, we have that

$$\begin{aligned}
V^{\mathcal{A}_t}_M(s_t) &\geqslant V^{\mathcal{A}_t}_M(s_t,H) \\
&\geqslant V^{\tilde{\pi}}_{M'}(s_t,H) - \Pr(A_M)/(1-\gamma) \\
&\geqslant V^{\tilde{\pi}}_{M'}(s_t,H) - \epsilon_1 \\
&\geqslant V^{\tilde{\pi}}_{M'}(s_t) - 2\epsilon_1 \\
&\geqslant V^{\tilde{\pi}}_{\tilde{M}}(s_t) - 3\epsilon_1 \\
&\geqslant V^*_M(s_t) - 3\epsilon_1.
\end{aligned}$$

The second step follows from Eq. (14). The fourth step follows from the definition of H. The fifth step follows from Eq. (13). For the sixth and final step, note that $V^{\tilde{\pi}}_{\tilde{M}}(s_t)$ is $\tilde{Q}(s_t, \tilde{\pi}(s_t))$, which is greater than $Q^*(s_t,a)$ for all a, by Lemma 6. Thus, it is also greater than $V^*(s_t)$.

Therefore, if $\epsilon_1 = \epsilon/3$, then MBIE's policy is ϵ-optimal with probability at least $1-\delta$ for all but $O(\frac{m|S||A|H}{\epsilon(1-\gamma)} \ln \frac{1}{\delta})$ many timesteps. Substitution of the values for m and H into this expression provides the desired result. □

4.3.3. Sample complexity of MBIE-EB

We start off by showing that MBIE-EB also exhibits "optimism in the face of uncertainty."

Lemma 7. *Suppose MBIE-EB is executed on any MDP M with $\beta = (1/(1-\gamma))\sqrt{\ln(2|S||A|m/\delta)/2}$. Then, with probability at least $1-\delta/2$, $\tilde{Q}(s,a) \geqslant Q^*(s,a)$ will always hold.*

[7] Consider two timesteps t_1 and t_2 with $t_1 < t_2 - H$. Technically, the event of escaping from K within H steps on or after timestep t_2 may not be independent of the same escape event on or after timestep t_1. However, the former event is conditionally independent of the later event given the history of the agent up to timestep t_2. Thus, we are able to apply Hoeffding's bound.

The proof of Lemma 7 appears in Appendix B. We now show that MBIE-EB is also PAC-MDP.

Theorem 2. *Suppose that ϵ and δ are two real numbers between 0 and 1 and $M = \langle S, A, T, \mathcal{R}, \gamma \rangle$ is any MDP. There exists an input $m = m(\frac{1}{\epsilon}, \frac{1}{\delta})$, satisfying $m(\frac{1}{\epsilon}, \frac{1}{\delta}) = O(\frac{|S|}{\epsilon^2(1-\gamma)^4} + \frac{1}{\epsilon^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon(1-\gamma)\delta})$, and $\beta = (1/(1-\gamma))\sqrt{\ln(2|S||A|m/\delta)/2}$ such that if MBIE-EB is executed on MDP M, then the following holds. Let $\mathcal{A}_t$ denote MBIE-EB's policy at time t and s_t denote the state at time t. With probability at least $1-\delta$, $V_M^{\mathcal{A}_t}(s_t) \geqslant V_M^*(s_t) - \epsilon$ is true for all but $O(\frac{|S||A|}{\epsilon^3(1-\gamma)^6}(|S| + \ln \frac{|S||A|}{\epsilon(1-\gamma)\delta}) \ln \frac{1}{\delta} \ln \frac{1}{\epsilon(1-\gamma)})$ timesteps t.*

Proof. Let ϵ_1 be an arbitrary positive real number, whose precise value we will specify later. Using Lemma 4, let $H = O(\frac{1}{1-\gamma} \ln \frac{1}{\epsilon_1(1-\gamma)})$ be a positive integer large enough so that for all MDPs M' with discount factor γ, policies π, and states s, $|V_{M'}^{\pi}(s, H) - V_{M'}^{\pi}(s)| \leqslant \epsilon_1$.

First, we argue that after each (s, a) has been experienced a polynomial number of times, m, the empirical model learned from those experiences, $\hat{R}(s, a)$ and $\hat{T}(s, a)$, will be sufficiently close to the true distributions, $\mathcal{R}(s, a)$ and $T(s, a)$, so that the value of any policy in any MDP model based on $\hat{R}$ and $\hat{T}$ is no more than ϵ_1 away from its value in the MDP based on R and T (but otherwise the same), with high probability. It follows from Lemma 2 that it is sufficient to require that $\|\hat{T}(s, a) - T(s, a)\|_1 \leqslant \tau$ and $|\hat{R}(s, a) - R(s, a)| \leqslant \tau$ for $\tau = C\epsilon_1(1-\gamma)^2$, where C is the constant specified in Lemma 2. Using the form of reward and confidence intervals used by MBIE with $\delta_R = \delta_T = \delta/(2|S||A|m)$, it follows that

$$m \geqslant C_1 \left(\frac{|S|}{\epsilon_1^2(1-\gamma)^4} + \frac{1}{\epsilon_1^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon_1(1-\gamma)\delta} \right) \tag{15}$$

for some positive constant C_1 is sufficient for these two conditions to hold, with probability at least $1 - \delta/2$. A formal proof of this is almost identical to the proof of Lemma 5, keeping in mind that while MBIE uses the upper tail of the transition and reward confidence intervals (see Section 3.1), the empirical model uses the center of these confidence intervals.[8]

Consider some fixed timestep t. Let s_t be the current state of the agent. Define K to be the set of all state-action pairs (s, a) that have been experienced at least m times ($n(s, a) = m$); we will provide the value of m shortly. We call K the set of *known state-action pairs*. Recall that the MBIE-EB agent (denoted by policy $\mathcal{A}_t$) chooses its next action by following an optimal policy $\tilde{\pi}$ of its internal model $\tilde{M}$ at time t. Let M' be the MDP that is equal to M on K (meaning equal reward and transition distributions for $(s, a) \in K$) and equal to $\tilde{M}$ on $S \times A - K$. Let $\hat{M}'$ be the MDP that is equal to $\hat{M}$ on K and equal to $\tilde{M}$ on $S \times A - K$. From our choice of m above, we have that

$$\left| V_{M'}^{\tilde{\pi}}(s) - V_{\hat{M}'}^{\tilde{\pi}}(s) \right| \leqslant \epsilon_1 \tag{16}$$

holds for all s, with probability at least $1 - \delta/2$. Also, note that $\tilde{M}$ is identical to the MDP $\hat{M}'$ except that some state-action pairs (precisely, those in K) have an additional reward bonus of $\beta/\sqrt{m}$. Thus, we have that $V_{\tilde{M}}^{\tilde{\pi}}(s) \leqslant V_{\hat{M}'}^{\tilde{\pi}}(s) + \beta/(\sqrt{m}(1-\gamma))$. For our analysis, we require that

$$\beta/\left(\sqrt{m}(1-\gamma)\right) \leqslant \epsilon_1. \tag{17}$$

We define

$$\beta = \left(1/(1-\gamma)\right)\sqrt{\ln\left(2|S||A|m/\delta\right)/2}. \tag{18}$$

It is not hard to show that we can make m large enough so that Eqs. (15) and (17) hold, yet small enough so that $m = O(\frac{|S|}{\epsilon_1^2(1-\gamma)^4} + \frac{1}{\epsilon_1^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon_1(1-\gamma)\delta})$. To see this fact, note that when substitution the value of β from Eq. (18) into Eq. (17) and simplifying, we arrive at

$$m \geqslant \frac{\ln(|S||A|m/\delta)/2}{(1-\gamma)^4 {\epsilon_1}^2}.$$

The rest follows from further basic algebraic manipulation. Given that Eq. (17) holds, we have that

$$V_{\tilde{M}}^{\tilde{\pi}}(s) \leqslant V_{\hat{M}'}^{\tilde{\pi}}(s) + \epsilon_1. \tag{19}$$

Let A_M be the event that $\tilde{\pi}$ "escapes" from K in H steps. Specifically, A_M is the event that some $(s, a) \notin K$ is experienced after following $\tilde{\pi}$ from state s_t for H-steps. We claim that for all states s:

$$V_M^{\mathcal{A}_t}(s, H) \geqslant V_{M'}^{\tilde{\pi}}(s, H) - \Pr(A_M)/(1-\gamma). \tag{20}$$

This follows from Lemma 3 and the fact that the policies $\mathcal{A}_t$ and $\tilde{\pi}$ behave identically unless A_M occurs.

We now consider two mutually exclusive cases. First, suppose that $\Pr(A_M) \geqslant \epsilon_1(1-\gamma)$, meaning that an agent following $\mathcal{A}_t$ will encounter an unknown (s, a) in H steps with probability at least $\epsilon_1(1-\gamma)$. Using the Hoeffding bound, after

[8] This difference amounts to a factor of 2 in the analysis.

$O(\frac{m|S||A|H}{\epsilon_1(1-\gamma)} \ln \frac{1}{\delta})$ timesteps t where $\Pr(A_M) \geqslant \epsilon_1(1-\gamma)$ is satisfied, all (s,a) will become known, with probability at least $1-\delta/2$.

Now, suppose that $\Pr(A_M) < \epsilon_1(1-\gamma)$. Then, we have that

$$\begin{aligned} V_M^{\mathcal{A}_t}(s_t) &\geqslant V_M^{\mathcal{A}_t}(s_t, H) \\ &\geqslant V_{M'}^{\tilde{\pi}}(s_t, H) - \Pr(A_M)/(1-\gamma) \\ &\geqslant V_{M'}^{\tilde{\pi}}(s_t, H) - \epsilon_1 \\ &\geqslant V_{M'}^{\tilde{\pi}}(s_t) - 2\epsilon_1 \\ &\geqslant V_{\hat{M}'}^{\tilde{\pi}}(s_t) - 3\epsilon_1 \\ &\geqslant V_{\hat{M}}^{\tilde{\pi}}(s_t) - 4\epsilon_1 \\ &\geqslant V_M^*(s_t) - 4\epsilon_1. \end{aligned}$$

The first step follows from the fact that all rewards are non-negative, so truncating the sum of expected rewards to only the next H rewards cannot increase the value function. The second step results from an application of Eq. (20). The third step follows from our assumption that $\Pr(A_M) < \epsilon_1(1-\gamma)$. The fourth step follows directly from how H was chosen. The fifth and sixth steps result from an application of Eq. (16) and Eq. (19), respectively. For the final step, note that $V_{\tilde{M}}^{\tilde{\pi}}(s_t)$ is $\tilde{Q}(s_t, \tilde{\pi}(s_t))$, which is greater than $Q^*(s_t, a)$ for all a by Lemma 7. Thus, it is also greater than $V^*(s_t)$.

Therefore, if $\epsilon_1 = \epsilon/4$, then MBIE-EB's policy is ϵ-optimal with probability at least $1-\delta$ for all but $O(\frac{m|S||A|H}{\epsilon(1-\gamma)} \ln \frac{1}{\delta})$ many timesteps. □

4.3.4. Discussion

Ignoring log factors, the bound given in Theorems 1 and 2 is

$$\tilde{O}\left(\frac{|S|^2|A|}{\epsilon^3(1-\gamma)^6}\right).$$

This bound matches the bound given by Kakade [10] in the analysis of the R-Max algorithm.[9] This is the best bound we could prove. In practice, improvement may be possible through the use of different types of bounds, such as the those based on KL-divergence. We also note that the best lower bound[10] known for the problem, due to Kakade [10], is $\tilde{\Omega}(|S||A|/(\epsilon(1-\gamma)))$.

Since the bounds we have proven for MBIE are comparable (in fact, slightly worse[11] asymptotically) to the bounds proven for R-Max, there is a question about whether it is really advantageous to use MBIE instead of R-Max. There is almost certainly a practical advantage if we focus on minimizing sample complexity rather than computation time. Informally, this is easy to argue. Note that it is possible to simulate any k-armed bandit problem on an MDP with a single state and k actions. The idea is to set the reward payoff distribution, for each action i, equal to the payoff distribution of arm i, for $i = 1, \ldots, k$. When we run MBIE and R-Max on the MDP formulation of a k-armed bandit problem, the algorithms behave like the IE and naïve algorithms, respectively, for bandit problems [7]. In [4], it was shown that an algorithm called *Successive Elimination* will perform much better than the naïve algorithm when there is a significant difference in the mean payoffs of different arms. Successive Elimination works much like IE in that confidence intervals are computed for each arm. However, successive elimination chooses each arm equally often (like naïve) until the upper tail of the confidence interval for some arm falls below that of another. At that point, the arm with smaller mean is eliminated and never tried again. It is immediately clear that IE will also eliminate such arms, assuming that the confidence intervals it computes are admissible. Thus, the bounds given by Even-Dar et al. [4] also apply to IE. So, for at least some MDPs, MBIE is provably better, in terms of sample complexity, than R-Max.

To formally explore the advantage of MBIE or MBIE-EB over R-Max, an analysis must quantify the sample complexity or average loss of the algorithms in terms of some vital parameters of the underlying MDP. We leave this as interesting future work.

---

[9] In that analysis, Kakade reported a bound of $\tilde{O}(\frac{|S|^2|A|}{\epsilon^3(1-\gamma)^3})$. However, he defined sample complexity in terms of a *normalized* value function. Using that definition, which reduces the dependence on $1/(1-\gamma)$, our bound is also $\tilde{O}(\frac{|S|^2|A|}{\epsilon^3(1-\gamma)^3})$.

[10] The bound from [10] was for the undiscounted setting. A slight modification yields the bound stated here.

[11] This is not surprising and appears in the PAC analysis of the naïve algorithm [4], which is related to the R-Max algorithm, and the IE algorithm [7], which is related to the MBIE algorithm, for the k-armed bandit problem. The logarithmic increase in sample complexity bound is due to an application of the Hoeffding bound over all intervals computed by IE rather than a single interval for naïve.

5. Average loss & sample complexity

A learning algorithm with low sample complexity will, with high probability, act with a near-optimal policy most of the time. On the other hand, a learning algorithm with low average loss is guaranteed to achieve near-optimal discounted reward from most of the encountered states with high probability. The latter is generally more desirable, since the final goal of reinforcement learning is to maximize reward. However, sample complexity analysis is cleaner and easier. Intuitively, it is clear that an agent minimizes its average loss precisely when it often acts with a near optimal policy, which is obtained by minimizing its sample complexity.

This section shows that a PAC algorithm in the sample-complexity framework is also PAC under average loss. Since average loss is arguably a more natural performance metric for learning in MDPs, while sample complexity admits a cleaner and more direct analysis, this result provides the best of both worlds.

5.1. Properties of adjusted average loss

Two properties of Definition 3 (instantaneous and average loss) present some bookkeeping difficulties. First, instantaneous loss compares the expected return of the optimal policy over an infinite length sequence ($V^*(s_t)$) to the return of the learning algorithm over a finite length path H. Second, the length of the finite sequence is variable and depends on the current timestep. The complexity of these properties is mitigated by the following definitions.

Definition 4. Suppose a learning algorithm is run for one sequence of $T_1 + T_2 - 1$ steps. Let c_t be the partial sequence $s_1, r_1, \ldots, s_{t-1}, r_{t-1}, s_t$. For any policy π and integer t such that $t \leqslant T_1$, let $R^{\pi}_{T_2}(t) := \sum_{t'=t}^{t+T_2-1} \gamma^{t'-t} r_{t'} + \gamma^{T_2} V^{\pi}(c_{t+T_2})$ be the **adjusted return**. Let $I^{\pi}_{T_2}(t) := V^{\pi}(c_t) - R^{\pi}_{T_2}(t)$ be the **adjusted instantaneous loss**. Let $L^{\pi}_{T_1,T_2} = \frac{1}{T_1} \sum_{t=1}^{T_1} I^{\pi}_{T_2}(t)$ be the **adjusted average loss**.

Adjusted return is the actual discounted sum of rewards starting at step t over the next T_2 steps, plus the discounted expected return π would receive starting from the state reached T_2 steps in the future. Adjusted instantaneous loss is the true return for policy π from time t minus the adjusted return—how much was lost relative to simply following π. Adjusted average loss is the average of the adjusted instantaneous losses over the first T_1 steps of the run. In these definitions, the policy π is not required to be the same as the policy followed by the algorithm.

For any (possibly non-stationary) policy π, MDP M, and integer H, we can run π in M for H steps. Let the partial sequence c_H be the list of states and rewards encountered by the agent along this run. Each time this experiment is performed, a different sequence might be generated. Thus, we say that c_H is *to be generated by* π, to emphasize the fact that c_H is a random partial sequence. Note that the adjusted instantaneous loss and adjusted average loss quantities are random variables dependent on the relevant partial sequence. We will find it useful to define the following additional random variables, $Y^{\pi}_t := V^{\pi}(c_t) - (r_t + \gamma V^{\pi}(c_{t+1}))$, for all $t < H$. In this definition, c_t is the partial sequence consisting of the prefix of c_H ending at state s_t (the tth state encountered). It follows from our definition that as long as the agent follows π, the expectation of Y^{π}_t is zero—it is the Bellman error in the value-function update for π.

Consider the sequence $Z := Y^{\pi}_1, Y^{\pi}_2, \ldots, Y^{\pi}_H$ of random variables up to time H. Next, we will show that any subsequence q of Z is a *martingale difference sequence*, meaning that each term in q has expectation zero even when conditioned on all previous terms of q.

Lemma 8. *Let π be a policy, and suppose the sequence $s_1, r_1, s_2, r_2, \ldots, s_H, r_H$ is to be generated by π. If $1 \leqslant q_1 < q_2 < \cdots < q_i < t \leqslant H$, then $E[Y^{\pi}_t | Y^{\pi}_{q_1}, Y^{\pi}_{q_2}, \ldots, Y^{\pi}_{q_i}] = 0$.*

Proof. Let $[Y^{\pi}_t | c_{t+1}]$ be the value of the random variable Y^{π}_t given the fixed partial sequence c_{t+1}. Then,

$$
\begin{aligned}
E\big[Y^{\pi}_t\big] &= \sum_{c_{t+1}} \Pr(c_{t+1}) \big[Y^{\pi}_t | c_{t+1}\big] \\
&= \sum_{c_t} \Pr(c_t) \sum_{r_t, s_{t+1}} \Pr(r_t, s_{t+1} | c_t) \big[Y^{\pi}_t | c_t, r_t, s_{t+1}\big].
\end{aligned}
$$

The sum in the first line above is over all possible sequences $c_{t+1} = s_1, r_1, \ldots, s_{t+1}$ resulting from t action choices by an agent following policy π.

In the term above, we note that conditioning Y^{π}_t on the sequence of random variables $Y^{\pi}_{q_1}, Y^{\pi}_{q_2}, \ldots, Y^{\pi}_{q_i}$ can certainly affect the probabilities $\Pr(c_t)$, by making some sequences more likely and others less likely. However, the term $\sum_{c_t} \Pr(c_t)$ will always be one. Notice that fixed values of $Y^{\pi}_{q_1}, Y^{\pi}_{q_2}, \ldots, Y^{\pi}_{q_i}$ cannot influence the innermost sum. Now, we have that

$$
\sum_{r_t, s_{t+1}} \Pr(r_t, s_{t+1} | c_t) \big[Y^{\pi}_t | c_t, r_t, s_{t+1}\big] = V^{\pi}(c_t) - \sum_{r_t, s_{t+1}} \Pr(r_t, s_{t+1} | c_t) \big(r_t + \gamma V^{\pi}(c_{t+1})\big).
$$

By the definition of $V^{\pi}(c_t)$, this last term is zero. □

Adjusted instantaneous loss can now be reformulated as the discounted sum of these random variables.

Lemma 9. *If $t \leqslant T_1$ is a positive integer, then $I^{\pi}_{T_2}(t) = \sum_{t'=t}^{t+T_2-1} \gamma^{t'-t} Y^{\pi}_{t'}$.*

Proof. Since only one policy π is considered, it is omitted from the notation:

$$\sum_{t'=t}^{t+T_2-1} \gamma^{t'-t} Y_{t'} = \sum_{t'=t}^{t+T_2-1} \gamma^{t'-t}\left[V(c_{t'}) - \left(r_{t'} + \gamma V(c_{t'+1})\right)\right]$$
$$= \left[V^{\pi}(c_t) - \gamma^{T_2} V^{\pi}(c_{t+T_2})\right] - \sum_{t'=t}^{t+T_2-1} \gamma^{t'-t} r_{t'}. \qquad \square$$

We will make use of the following well-known result.

Lemma 10 *(Azuma's Lemma). If the random variables $X_1, X_2, \ldots$ form a* martingale difference sequence, *meaning that $E[X_k | X_1, X_2, \ldots, X_{k-1}] = 0$ for all k, and $|X_k| \leqslant b$ for each k, then*

$$P\left[\sum_{t=1}^{k} X_t > a\right] \leqslant \exp\left(-\frac{a^2}{2b^2 k}\right).$$

5.2. Adjusted and average loss

This section shows that the quantities T_1 and T_2, the number and length of the trials in Definition 4, may be only polynomially large and still ensure that results about adjusted loss apply to average loss. In the following, recall that we have assumed all rewards are between 0 and 1.

Lemma 11. *For any $0 < \epsilon < 1$, T_2 can be chosen so that $il(t) - I^{\pi^*}_{T_2}(t) \leqslant \epsilon$.*

Proof. Let $H = T_1 + T_2 - 1$, then

$$il(t) - I^{\pi^*}_{T_2}(t) = \gamma^{T_2} V^*(s_{t+T_2}) - \sum_{i=t+T_2}^{H} \gamma^{i-t} r_i$$
$$\leqslant \gamma^{T_2} V^*(s_{t+T_2}) \leqslant \gamma^{T_2} \frac{1}{1-\gamma}.$$

Now, simply set $T_2 \geqslant \frac{\ln(\epsilon(1-\gamma))}{\ln(\gamma)}$. $\square$

Lemma 11 shows that the adjusted instantaneous loss is not far from the instantaneous loss.

Proposition 3. *Suppose that $l \geqslant 0$. If $T_1 \geqslant \frac{2T_2}{\epsilon}$ and $T_2 \geqslant \frac{\ln(\epsilon(1-\gamma)}{2)}/\ln(\gamma)$, then $l - L^{\pi^*}_{T_1,T_2} \leqslant \epsilon$.*

Proof.

$$l - L^{\pi^*}_{T_1,T_2} = \frac{1}{H}\sum_{t=1}^{H} il(t) - \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\pi^*}_{T_2}(t)$$
$$\leqslant \frac{1}{T_1}\sum_{t=1}^{T_1}\left(il(t) - I^{\pi^*}_{T_2}(t)\right) + \frac{1}{T_1}\sum_{t=T_1+1}^{T_1+T_2-1} il(t)$$
$$\leqslant \frac{1}{T_1}\sum_{t=1}^{T_1}\epsilon + \left(\frac{T_2}{T_1}\right) = \epsilon + \left(\frac{T_2}{T_1}\right).$$

The second step above follows from the fact that $l \geqslant 0$ and $T_1 < H$. Lemma 11 is applied in the third step. The result now follows by letting $T_1 \geqslant \frac{T_2}{\epsilon}$. $\square$

The importance of the result is that we can bound the average loss l, by bounding the adjusted loss $L^{\pi}_{T_1,T_2}$.

5.3. Reduction to sample complexity

Our main objective is to relate sample complexity and average loss. We now show that the number of trials T_1 used in the adjusted definition of average loss can be made large enough (but not more than polynomially large) so that, with high probability, any algorithm's average loss can be made arbitrarily small given that the algorithm's sample complexity is bounded with high probability.

Proposition 4. *Suppose T_2 and C are two positive integers. If C is a bound on the sample complexity of some algorithm $\mathcal{A}$ with respect to ϵ, which holds with probability at least $1-\delta$, then T_1 can be chosen so that the adjusted average loss satisfies $L^{\pi^*}_{T_1,T_2} \leqslant 3\epsilon$, with probability at least $1-2\delta$.*

Proof. Consider running algorithm $\mathcal{A}$, which can be viewed as a non-stationary policy, in the MDP M for $H := T_1 + T_2 - 1$ steps. We have that

$$L^{\mathcal{A}}_{T_1,T_2} = \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t). \tag{21}$$

The left side of this equation is the adjusted average loss of $\mathcal{A}$ *with respect to itself*. Intuitively, we expect this quantity to be very small provided that T_1 is sufficiently large. We now formally verify this intuition.

We examine the terms, $I^{\mathcal{A}}_{T_2}(t)$, of Eq. (21). Let $v_{\max}$ denote the quantity $1/(1-\gamma)$. We have that $E[I^{\mathcal{A}}_{T_2}(t)] = 0$, since $\mathcal{A}$ is the policy that produces the trajectory used to compute $I^{\mathcal{A}}_{T_2}(t)$. However, note that the terms are not pairwise independent. This is because $I^{\mathcal{A}}_{T_2}(t)$, for various values of t, are computed using the same rewards (from the trajectory generated by executing $\mathcal{A}$ in the MDP M).

Now, consider the sum of Eq. (21):

$$\begin{aligned}
\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t) &= \sum_{t=1}^{T_1}\sum_{t'=t}^{t+T_2-1} \gamma^{t'-t} Y^{\mathcal{A}}_{t'} \\
&= \sum_{t'=1}^{T_2-1}\sum_{t=1}^{t'} \gamma^{t'-t} Y^{\mathcal{A}}_{t'} + \sum_{t'=T_2}^{T_1+T_2+1}\sum_{t=t'-T_2+1}^{T_1} \gamma^{t'-t} Y^{\mathcal{A}}_{t'} \\
&= \sum_{t'=1}^{T_2-1} Y^{\mathcal{A}}_{t'}\sum_{t=1}^{t'} \gamma^{t'-t} + \sum_{t'=T_2}^{T_1+T_2+1} Y^{\mathcal{A}}_{t'}\sum_{t=t'-T_2+1}^{T_1} \gamma^{t'-t}.
\end{aligned}$$

The second equality above results from switching the order of the summands, which allows us to evaluate the innermost sums of that line. The last line reveals that $\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t)$ is the sum of a martingale difference sequence, where each term is bounded by $v_{\max}/(1-\gamma)$. To see this fact, note that the term $Y^{\mathcal{A}}_{t'} := V^{\mathcal{A}}(c_t) - (r_t + \gamma V^{\mathcal{A}}(c_{t+1})) \leqslant 1/(1-\gamma) = v_{\max}$ and $\sum_t \gamma^{t'-t} \leqslant 1/(1-\gamma)$. Therefore, applying Azuma's Lemma yields

$$P\left(\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t) > a\right) \leqslant \exp\left(\frac{-a^2(1-\gamma)^2}{2{v_{\max}}^2(T_1+T_2-1)}\right). \tag{22}$$

Thus, we have that

$$\frac{1}{T_1}\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t) \leqslant \epsilon \tag{23}$$

holds with probability at least $1-\delta$ when $\exp(\frac{-T_1^2\epsilon^2(1-\gamma)^2}{2v_{\max}^2(T_1+T_2-1)}) \leqslant \delta$. This condition is equivalent to the following:

$$T_1\left(T_1\epsilon^2(1-\gamma)^2 - 2\ln(1/\delta)v_{\max}^2\right) \geqslant 2\ln(1/\delta)v_{\max}^2(T_2-1). \tag{24}$$

Eq. (24) is satisfied when the following holds:

$$T_1 \geqslant \max\left\{\frac{1+2\ln(1/\delta)v_{\max}^2}{\epsilon^2(1-\gamma)^2}, 2\ln(1/\delta)v_{\max}^2(T_2-1)\right\}. \tag{25}$$

The next step is to relate $L^{\mathcal{A}}_{T_1,T_2}$ to $L^{\pi^*}_{T_1,T_2}$, the adjusted average loss with respect to the optimal value function. Partition the generated partial sequence $s_1, r_1, \ldots, s_{T_1}, r_{T_1}$, into those timesteps $t \in S_B$ such that $\mathcal{A}$ is not ϵ-optimal, and those timesteps $t \in S_G$ such that it is. We have that

$$
\begin{aligned}
L^{\pi^*}_{T_1,T_2} &= \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\pi^*}_{T_2}(t) \\
&= \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\pi^*}_{T_2}(t) + \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t) - \frac{1}{T_1}\sum_{t=1}^{T_1} I^{\mathcal{A}}_{T_2}(t) \\
&\leqslant \epsilon + \frac{1}{T_1}\sum_{t=1}^{T_1}\left(I^{\pi^*}_{T_2}(t) - I^{\mathcal{A}}_{T_2}(t)\right) \\
&= \epsilon + \frac{1}{T_1}\sum_{t\in S_B}\left(I^{\pi^*}_{T_2}(t) - I^{\mathcal{A}}_{T_2}(t)\right) + \frac{1}{T_1}\sum_{t\in S_G}\left(I^{\pi^*}_{T_2}(t) - I^{\mathcal{A}}_{T_2}(t)\right) \\
&\leqslant \epsilon + \frac{1}{T_1}(C)(v_{\max}) + \frac{1}{T_1}\sum_{t\in S_G}\left(I^{\pi^*}_{T_2}(t) - I^{\mathcal{A}}_{T_2}(t)\right) \\
&\leqslant \epsilon + \frac{1}{T_1}(C)(v_{\max}) + \frac{1}{T_1}\sum_{t\in S_G}\left(V^{\pi^*}(c_t) - R^{\pi^*}_{T_2}(t) - \left(V^{\mathcal{A}}(c_t) - R^{\mathcal{A}}_{T_2}(t)\right)\right) \\
&\leqslant 2\epsilon + \frac{1}{T_1}(C)(v_{\max}) + \frac{1}{T_1}\sum_{t\in S_G}\left(R^{\mathcal{A}}_{T_2}(t) - R^{\pi^*}_{T_2}(t)\right) \\
&\leqslant 2\epsilon + \frac{1}{T_1}(C)(v_{\max}).
\end{aligned}
$$

In the second step we added and subtracted the same term. In the third step we made use of Eq. (23). In the forth step we partitioned the sum into those terms with $t \in S_B$ and $t \in S_G$. In the fifth step, we used the precondition of the theorem ($|S_B| \leqslant C$) with the fact that $I^{\pi^*}_{T_2}(t) - I^{\mathcal{A}}_{T_2}(t) \leqslant v_{\max}$. The sixth step follows from Definition 4. The seventh step follows from the fact that $V^{\mathcal{A}}(c_t) \geqslant V^{\pi^*}(c_t) - \epsilon$ for $t \in S_G$. The final step follows from examining the definition of $R^{\pi^*}_{T_2}(t)$ and noting that the value of the policy π^* is never less than the value of $\mathcal{A}$.

We have shown that $L^{\pi^*}_{T_1,T_2} \leqslant 2\epsilon + \frac{1}{T_1}(C)(v_{\max})$, with probability at least $1-2\delta$. To ensure that the second term of this expression is no more than ϵ, it is sufficient to enforce the following inequality:

$$T_1 \geqslant \frac{1}{\epsilon}(C)(v_{\max}). \tag{26}$$

Note that T_1 can satisfy Eqs. (25) and (26), yet still be no more than polynomial in the relevant quantities $1/\delta$, $1/\epsilon$, $1/(1-\gamma)$, C, and T_2. □

In summary, low sample complexity implies low average loss since the algorithm does not have to run too long before the number of near-optimal trials is sufficient to make the average loss low.

6. Experiments

In this section, we present a comparison of the two variants of MBIE to two other algorithms, E^3 [11] and R-Max [3], for the purpose of validating the basic MBIE approach.

We report results for two simple MDPs, both taken from [19]. We measure the cumulative reward over the first 5000 steps of learning. The algorithms seek to maximize discounted reward ($\gamma = .95$). Each algorithm uses value iteration to solve its model. Value iteration is terminated when updates to each state-action pair result in a change less than 0.01.

Each of the three algorithms has one or two principle parameters that, coarsely, trades off between exploration and exploitation. To better quantify the effect of the parameters, we repeated experiments with a range of parameter settings and reported the best setting. For both MBIE variants, we placed no limit on the model size (equivalent to setting $m = \infty$ in Section 3). The MBIE algorithm has two parameters A and B. Recall that MBIE computes two confidence intervals, reward and transition, for each state-action pair (s,a). The widths of these confidence intervals are $\epsilon^R_{n(s,a)}$ and $\epsilon^T_{n(s,a)}$, respectively (see Sections 3.1.1 and 3.1.2), where $n(s,a)$ is the number of visits to (s,a). Let $R_{\max}$ denote the maximum achievable reward. We parameterized these two intervals in the following way

$$\epsilon^R_{n(s,a)} = A\frac{R_{\max}}{\sqrt{n(s,a)}}, \tag{27}$$

$$\epsilon^T_{n(s,a)} = B\frac{1}{\sqrt{n(s,a)}}. \tag{28}$$

The MBIE-EB algorithm has one parameter C, which corresponds to $\beta/R_{\max}$ in Eq. (8). The R-Max algorithm has a single parameter m and E^3 has two parameters m and *thresh*. The parameter m for both R-Max and E^3 specifies the number of

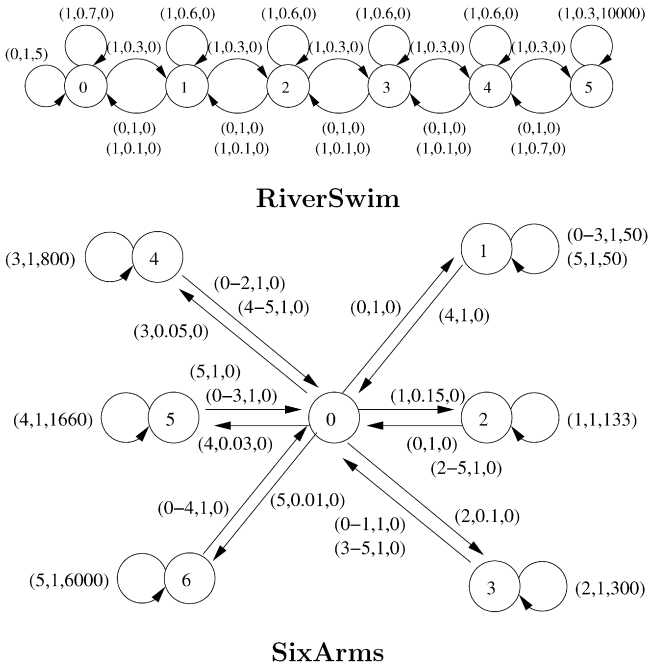


Fig. 1. Two example environments: **RiverSwim** (top), and **SixArms** (bottom). Each node in the graph is a state and each edge is labeled by one or more transitions. A transition is of the form (a, p, r) where a is an action, p the probability that action will result in the corresponding transition, and r is the reward for taking the transition. For **SixArms**, the agent starts in state 0. For **RiverSwim**, the agent starts in either state 1 or 2 with equal probability.

times a state-action pair must be visited before the experience for that state-action pair is utilized in the agent's model.[12] The E^3 algorithm computes two policies during each step, an exploration policy and an exploitation policy. It also computes an approximation to the probability that the exploration policy will reach a state-action pair that has not been experienced m times. If this approximated probability is larger than *thresh* it executes the exploration policy. Otherwise, it executes the exploitation policy.

The two MDPs we used in our experiments are illustrated in Fig. 1. **RiverSwim** consists of six states. The agent starts in one of the states near the beginning of the row. The two actions available to the agent are to swim left or right. Swimming to the right (against the current of the river) will more often than not leave the agent in the same state, but will sometimes transition the agent to the right (and with a much smaller probability to the left). Swimming to the left (with the current) always succeeds in moving the agent to the left, until the leftmost state is reached at which point swimming to the left yields a small reward of five units. The agent receives a much larger reward, of ten thousand units, for swimming upstream and reaching the rightmost state. This MDP requires sequences of actions in order to explore effectively.

SixArms consists of seven states, one of which is the initial state. The agent must choose from six different actions. Each action pulls a different arm, with a different payoff probability, on a k-armed bandit. When the arm pays off, the agent is sent to another state. Within that new state, large rewards can be obtained. The higher the payoff probability for an arm, the smaller the reward received (from the room the agent is sent to by the arm). In this MDP, a decision maker can make use of smaller amounts of experience on the low paying arms and still perform well.

Exploration is important in both domains as Strehl and Littman [19] show that a simple model-based algorithm that uses ϵ-greedy exploration fails to learn quickly on them. The results of our experiments are shown in Figs. 2 and 3. In **RiverSwim**, all four algorithms that we tested are effective at using focused exploration to ferret out the high-reward state by sequencing exploration steps. In **SixArms**, the MBIE variants are quickly able to rule out low-reward actions, resulting in near optimal behavior. For R-Max and E^3, however, low exploration settings result in premature convergence to a suboptimal policy while high values result in many wasted trials on the clearly suboptimal actions.

7. Conclusion

We have shown that Interval Estimation can be incorporated with model-based reinforcement learning to guide exploration. The resulting algorithm, MBIE, then satisfies a PAC-like guarantee on the number of times it does not behave near optimally. We have also introduced a similar, yet simpler, model-based algorithm called MBIE-EB that has the same guarantees but is easier to implement. In addition, we considered a more "online" performance metric called average loss that

[12] Before m samples are obtained for a given state-action pair, its value is maximally rewarding in R-Max's model and in one of E^3's two models.

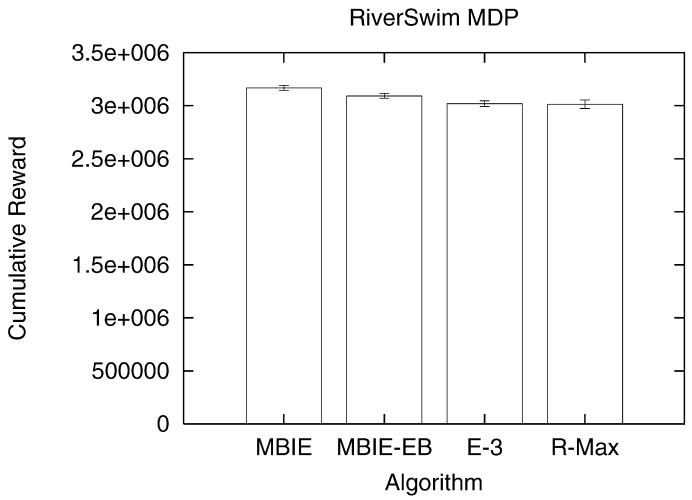


Fig. 2. A comparison of the algorithms on the **RiverSwim** MDP with parameter settings (best found during a broad search) $A = 0.3$ and $B = 0$ for MBIE; $C = 0.4$ for MBIE-EB; $m = 16$ for R-Max; and $m = 16$ and $thresh = 0.01$ for E^3.

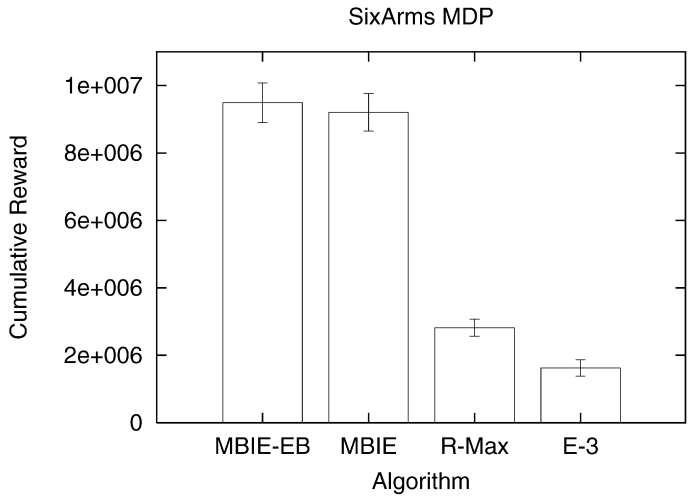


Fig. 3. A comparison of the algorithms on the **SixArms** MDP with parameter settings (best found during a broad search) $C = 0.8$ for MBIE-EB; $A = 0.3$ and $B = 0.08$ for MBIE; $m = 6$ for R-Max; and $m = 4$ and $thresh = 0.09$ for E^3.

takes the actual rewards received by an agent into account. Theoretically, we have related it, in a PAC sense, to another more standard performance metric called the sample complexity of exploration.

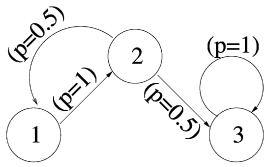
An important direction of future work is to generalize the results to MDPs with infinite state and action spaces.

Acknowledgments

Thanks to the National Science Foundation (IIS-0325281) for funding. Csaba Szepesvári and Martin Zinkovich were particularly helpful with editing and clarifying some of the technical content. We also thank Sanjoy Dasgupta, John Langford, Lihong Li, Shie Mannor, David McAllester, Ali Nouri, Rob Schapire, Rich Sutton, Hado van Hasselt, Sergio Verdu, Thomas Walsh, Tsachy Weissman, and Eric Wiewiora for suggestions.

Appendix A. Independence of samples

Suppose that we wish to estimate the transition probability of reaching a fixed state s' after experiencing a fixed state-action pair (s, a). We require an ϵ-accurate estimate with probability at least $1 - \delta$, for some predefined values ϵ and δ. Let D be the distribution that produces a 1 if s' is reached after experiencing (s, a) and 0 otherwise. Using the Hoeffding bound we can compute a number m, polynomial in $1/\epsilon$ and $1/\delta$, so that m independent samples of D can be averaged and used as an estimate $\hat{T}(s'|s, a)$. To obtain these samples, we must wait until the agent reaches state s and takes action a at least m times. Unfortunately, the dynamics of the MDP may exist so that the event of reaching state s at least m times



An example MDP.

Fig. 4.

provides information about which m samples were obtained from experiencing (s,a). For example, consider the MDP of Fig. 4. There are 3 states and a single action. Under action 1, state 1 leads to state 2; state 2 leads, with equal probability, to state 1 and state 3; and state 3 leads to itself. Thus, once the agent is in state 3 it cannot reach state 2. Suppose we would like to estimate the probability of reaching state 1 from state 2. After our mth experience of state 2, our estimated probability will be either 1 or $(m-1)/m$, both of which are very far from the true probability of $1/2$. This happens because the samples are not independent.

Fortunately, this issue is resolvable, and we can essentially assume that the samples are independent. The key observation is that in the example of Fig. 4, the probability of reaching state 2 at least m times is also extremely low. It turns out that the probability that an agent (following any policy) observes any fixed m samples of next-states after experiencing (s,a) is at most the probability of observing those same m samples after m independent draws from the transition distribution T. We formalize this now.

Consider a fixed state-action pair (s,a). Upon execution of a learning algorithm on an MDP, we consider the (possibly finite) sequence $O_{s,a} = [O_{s,a}(i)]$, where $O_{s,a}(i)$ is an ordered pair containing the next-state and immediate reward that resulted from the ith experience of (s,a). Let $\mathcal{Q} = [(s[1],r[1]),\ldots,(s[m],r[m])] \in (|S|\times\mathbb{R})^m$ be any finite sequence of m state and reward pairs. Next, we upper bound the probability that the first m elements of $O_{s,a}$ match $\mathcal{Q}$ exactly.

Claim C1. *For a fixed state-action pair (s,a), the probability that the sequence $\mathcal{Q}$ is observed by the learning agent (meaning that m experiences of (s,a) do occur and each next-state and immediate reward observed after experiencing (s,a) matches exactly the sequence in $\mathcal{Q}$) is at most the probability that $\mathcal{Q}$ is obtained by a process of drawing m random next-states and rewards from distributions $T(s,a)$ and $\mathcal{R}(s,a)$, respectively. The claim is a consequence of the Markov property.*

Proof. Let $s(i)$ and $r(i)$ denote the (random) next-state reached and immediate reward received on the ith experience of (s,a), for $i=1,\ldots,m$ (where $s(i)$ and $r(i)$ take on special values $\emptyset$ and -1, respectively, if no such experience occurs). Let $Z(i)$ denote the event that $s(j)=s[j]$ and $r(j)=r[j]$ for $j=1,\ldots,i$. Let $W(i)$ denote the event that (s,a) is experienced at least i times. We want to bound the probability that event $Z := Z(m)$ occurs (that the agent observes the sequence $\mathcal{Q}$). We have that

$$\Pr[Z] = \Pr\big[s(1)=s[1]\wedge r(1)=r[1]\big]\cdots\Pr\big[s(m)=s[m]\wedge r(m)=r[m]|Z(m-1)\big]. \tag{A.1}$$

For the ith factor of the right-hand side of Eq. (A.1), we have that

$$\begin{aligned}\Pr\big[s(i)=s[i]\wedge r(i)=r[i]|Z(i-1)\big] &= \Pr\big[s(i)=s[i]\wedge r(i)=r[i]\wedge W(i)|Z(i-1)\big]\\ &= \Pr\big[s(i)=s[i]\wedge r(i)=r[i]|W(i)\wedge Z(i-1)\big]\Pr\big[W(i)|Z(i-1)\big]\\ &= \Pr\big[s(i)=s[i]\wedge r(i)=r[i]|W(i)\big]\Pr\big[W(i)|Z(i-1)\big].\end{aligned}$$

The first step follows from the fact that $s(i)=s[i]$ and $r(i)=r[i]$ can only occur if (s,a) is experienced for the ith time (event $W(i)$). The last step is a consequence of the Markov property. In words, the probability that the ith experience of (s,a) (if it occurs) will result in next-state $s[i]$ and immediate reward $r[i]$ is conditionally independent of the event $Z(i-1)$ given that (s,a) is experienced at least i times (event $W(i)$). Using the fact that probabilities are at most 1, we have shown that $\Pr[s(i)=s[i]\wedge r(i)=r[i]|Z(i-1)] \leqslant \Pr[s(i)=s[i]\wedge r(i)=r[i]|W(i)]$ Hence, we have that

$$\Pr[Z] \leqslant \prod_{i=1}^{m}\Pr\big[s(i)=s[i]\wedge r(i)=r[i]|W(i)\big].$$

The right-hand side, $\prod_{i=1}^{m}\Pr[s(i)=s[i]\wedge r(i)=r[i]|W(i)]$ is the probability that $\mathcal{Q}$ is observed after drawing m random next-states and rewards (as from a generative model for MDP M). □

To summarize, we may assume the samples are independent if we only use this assumption when upper bounding the probability of certain sequences of next-states or rewards. This is valid because, although the samples may not be independent, any upper bound that holds for independent samples also holds for samples obtained in an online manner by the agent.

Appendix B. Proofs

Proof of Lemma 1. Let $\Delta := \max_{(s,a)\in S\times A} |Q_1^\pi(s,a) - Q_2^\pi(s,a)|$. Let π be a fixed policy and (s,a) be a fixed state-action pair. We overload notation and let R_i denote $R_i(s,a)$, $T_i(s')$ denote $T_i(s'|s,a)$, and $V_i^\pi(s')$ denote $Q_i^\pi(s',\pi(s'))$ for $i=1,2$. We have that

$$\begin{aligned}
\left|Q_1^\pi(s,a) - Q_2^\pi(s,a)\right| &= \left|R_1 + \gamma\sum_{s'\in S} T_1(s')V_1^\pi(s') - R_2 - \gamma\sum_{s'\in S} T_2(s')V_2^\pi(s')\right| \\
&\leqslant |R_1 - R_2| + \gamma\left|\sum_{s'\in S}\left[T_1(s')V_1^\pi(s') - T_2(s')V_2^\pi(s')\right]\right| \\
&\leqslant \alpha + \gamma\left|\sum_{s'\in S}\left[T_1(s')V_1^\pi(s') - T_1(s')V_2^\pi(s') + T_1(s')V_2^\pi(s') - T_2(s')V_2^\pi(s')\right]\right| \\
&\leqslant \alpha + \gamma\left|\sum_{s'\in S} T_1(s')\left[V_1^\pi(s') - V_2^\pi(s')\right]\right| + \gamma\left|\sum_{s'\in S}\left[T_1(s') - T_2(s')\right]V_2^\pi(s')\right| \\
&\leqslant \alpha + \gamma\Delta + \frac{\gamma R_{\max}\beta}{(1-\gamma)}.
\end{aligned}$$

The first step used Bellman's equation. The second and fourth steps used the triangle inequality. In the third step, we added and subtracted the term $T_1(s')V_2^\pi(s')$. In the fifth step we used the bound on the L1 distance between the two transition distributions and the fact that all value functions are bounded by $R_{\max}/(1-\gamma)$. We have shown that $\Delta \leqslant \alpha + \gamma\Delta + \frac{\gamma R_{\max}\beta}{(1-\gamma)}$. Solving for Δ yields the desired result. □

Proof of Lemma 7. First, for some state-action pair (s,a), consider the first $k \leqslant m$ experiences of (s,a) by the agent (timesteps for which action a was taken from state s). Let $X_1, X_2, \ldots, X_k$ be the k random variables defined by:

$$X_i := r_i + \gamma V^*(s_i) \tag{B.1}$$

where r_i is the ith reward received and s_i is the ith state visited after taking action a from state s. Note that $E[X_i] = Q^*(s,a)$ and that $0 \leqslant X_i \leqslant 1/(1-\gamma)$ for all $i = 1,\ldots,k$. Assuming that the X_i are independently and identically distributed (see Appendix A), we can apply the Hoeffding bound to arrive at

$$\Pr\left[E[X_1] - (1/k)\sum_{i=1}^{k} X_i \geqslant \beta/\sqrt{k}\right] \leqslant e^{-2\beta^2(1-\gamma)^2}.$$

The value of β specified by the lemma guarantees that the right-hand side above is $\delta/(2|S||A|m)$. Note that $(1/k)\sum_{i=1}^{k} X_i = \hat{R}(s,a) + \gamma\sum_{s'}\hat{T}(s'|s,a)V^*(s')$. Once (s,a) has been experienced m times, $\hat{R}$ and $\hat{T}$ cannot change again. Therefore, by the union bound we have that

$$\hat{R}(s,a) + \gamma\sum_{s'}\hat{T}(s'|s,a)V^*(s') - Q^*(s,a) \geqslant -\beta/\sqrt{k} \tag{B.2}$$

holds for all timesteps t and all state-action pairs (s,a) with probability at least $1-\delta/2$, where $k = n_t(s,a)$.

Fix a timestep t. Recall that MBIE-EB uses value iteration to solve the following set of equations:

$$\tilde{Q}(s,a) = \hat{R}(s,a) + \gamma\sum_{s'}\hat{T}(s'|s,a)\max_{a'}\tilde{Q}(s',a') + \frac{\beta}{\sqrt{n(s,a)}}. \tag{B.3}$$

The proof is by induction on the number of steps of value iteration.[13] Let $\tilde{Q}^{(i)}(s,a)$ denote the ith iterate of value iteration for (s,a), and let $\tilde{V}^{(i)}(s) = \max_a \tilde{Q}^{(i)}(s,a)$. For the base case, by optimistic initialization we have that $\tilde{Q}^{(0)} = 1/(1-\gamma) \geqslant Q^*(s,a)$ for all state-action pairs (s,a). Now, for the induction, suppose that the claim holds for the current value function $\tilde{Q}^{(i)}(s,a)$. We have that

$$\begin{aligned}
\tilde{Q}^{(i+1)}(s,a) &= \hat{R}(s,a) + \gamma\sum_{s'}\hat{T}(s'|s,a)\max_{a'}\tilde{Q}^{(i)}(s',a') + \frac{\beta}{\sqrt{n(s,a)}} \\
&= \hat{R}(s,a) + \gamma\sum_{s'}\hat{T}(s'|s,a)\tilde{V}^{(i)}(s') + \frac{\beta}{\sqrt{n(s,a)}}
\end{aligned}$$

[13] We assume here that value iteration is halted after a finite number of iterations.

$$\geqslant \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s'|s,a) V^*(s') + \frac{\beta}{\sqrt{n(s,a)}}$$
$$\geqslant Q^*(s,a).$$

The first two steps follow from definitions. In the last two steps we used the inductive hypothesis and Eq. (B.2), respectively. □

References

[1] P. Auer, Using confidence bounds for exploitation–exploration trade-offs, J. Mach. Learn. Res. 3 (2002) 397–422.
[2] P. Auer, R. Ortner, Online regret bounds for a new reinforcement learning algorithm, in: First Austrian Cognitive Vision Workshop, 2005, pp. 35–42.
[3] R.I. Brafman, M. Tennenholtz, R-MAX—A general polynomial time algorithm for near-optimal reinforcement learning, J. Mach. Learn. Res. 3 (2002) 213–231.
[4] E. Even-Dar, S. Mannor, Y. Mansour, PAC bounds for multi-armed bandit and Markov decision processes, in: 15th Annual Conference on Computational Learning Theory (COLT), 2002, pp. 255–270.
[5] E. Even-Dar, S. Mannor, Y. Mansour, Action elimination and stopping conditions for reinforcement learning, in: The Twentieth International Conference on Machine Learning (ICML 2003), 2003, pp. 162–169.
[6] C.-N. Fiechter, Expected mistake bound model for on-line reinforcement learning, in: Proceedings of the Fourteenth International Conference on Machine Learning, 1997, pp. 116–124.
[7] P.W.L. Fong, A quantitative study of hypothesis selection, in: Proceedings of the Twelfth International Conference on Machine Learning (ICML-95), 1995, pp. 226–234.
[8] R. Givan, S. Leach, T. Dean, Bounded-parameter Markov decision processes, Artificial Intelligence 122 (1–2) (2000) 71–109.
[9] L.P. Kaelbling, Learning in Embedded Systems, The MIT Press, Cambridge, MA, 1993.
[10] S.M. Kakade, On the sample complexity of reinforcement learning, PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.
[11] M.J. Kearns, S.P. Singh, Near-optimal reinforcement learning in polynomial time, Machine Learning 49 (2–3) (2002) 209–232.
[12] M.J. Kearns, U.V. Vazirani, An Introduction to Computational Learning Theory, The MIT Press, Cambridge, MA, 1994.
[13] T.L. Lai, Adaptive treatment allocation and the multi-armed bandit problem, Ann. Statist. 15 (3) (1987) 1091–1114.
[14] A. Nilim, L.E. Ghaoui, Robustness in Markov decision problems with uncertain transition matrices, in: Advances in Neural Information Processing Systems 16 (NIPS-03), 2004.
[15] M.L. Puterman, Markov Decision Processes—Discrete Stochastic Dynamic Programming, John Wiley & Sons, Inc., New York, NY, 1994.
[16] M.J. Streeter, S.F. Smith, A simple distribution-free approach to the max k-armed bandit problem, in: CP 2006: Proceedings of the Twelfth International Conference on Principles and Practice of Constraint Programming, 2006.
[17] A.L. Strehl, L. Li, M.L. Littman, Incremental model-based learners with formal learning-time guarantees, in: UAI-06: Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence, 2006, pp. 485–493.
[18] A.L. Strehl, L. Li, E. Wiewiora, J. Langford, M.L. Littman, PAC model-free reinforcement learning, in: ICML-06: Proceedings of the 23rd International Conference on Machine Learning, 2006, pp. 881–888.
[19] A.L. Strehl, M.L. Littman, An empirical evaluation of interval estimation for Markov decision processes, in: The 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-2004), 2004, pp. 128–135.
[20] A.L. Strehl, M.L. Littman, A theoretical analysis of model-based interval estimation, in: Proceedings of the Twenty-Second International Conference on Machine Learning (ICML-05), 2005, pp. 857–864.
[21] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, The MIT Press, 1998.
[22] L.G. Valiant, A theory of the learnable, Comm. ACM 27 (11) (1984) 1134–1142.
[23] T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, M.J. Weinberger, Inequalities for the L1 deviation of the empirical distribution, Tech. Rep. HPL-2003-97R1, Hewlett-Packard Labs, 2003.
[24] M. Wiering, J. Schmidhuber, Efficient model-based exploration, in: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior (SAB'98), 1998, pp. 223–228.
[25] J.L. Wyatt, Exploration control in reinforcement learning using optimistic model selection, in: Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), 2001, pp. 593–600.