Near-Optimal Reinforcement Learning in Polynomial Time

Michael Kearns*
Syntek Capital
New York, NY 10019
mkearns@cis.upenn.edu

Satinder Singh
Syntek Capital
New York, NY 10019
baveja@cs.colorado.edu

January 1998

Abstract

We present new algorithms for reinforcement learning and prove that they have polynomial bounds on the resources required to achieve near-optimal return in general Markov decision processes. After observing that the number of actions required to approach the optimal return is lower bounded by the mixing time T of the optimal policy (in the undiscounted case) or by the horizon time T (in the discounted case), we then give algorithms requiring a number of actions and total computation time that are only polynomial in T and the number of states and actions, for both the undiscounted and discounted cases. An interesting aspect of our algorithms is their explicit handling of the Exploration-Exploitation trade-off.

*This work was done while both authors were at AT&T labs.

1 Introduction

In reinforcement learning, an agent interacts with an unknown environment, and attempts to choose actions that maximize its cumulative payoff (Sutton & Barto, 1998; Barto et al., 1990; Bertsekas & Tsitsiklis, 1996). The environment is typically modeled as a Markov decision process (MDP), and it is assumed that the agent does not know the parameters of this process, but has to *learn* how to act directly from experience. Thus, the reinforcement learning agent faces a fundamental trade-off between *exploitation* and *exploration* (Bertsekas, 1987; Kumar & Varaiya, 1986; Thrun, 1992): that is, should the agent exploit its cumulative experience so far, by executing the action that currently seems best, or should it execute a different action, with the hope of gaining information or experience that could lead to higher future payoffs? Too little exploration can prevent the agent from ever converging to the optimal behavior, while too much exploration can prevent the agent from gaining near-optimal payoff in a timely fashion.

There is a large literature on reinforcement learning, which has been growing rapidly in the last decade. Many different algorithms have been proposed to solve reinforcement learning problems, and various theoretical results on the convergence properties of these algorithms have been proven. For example, Watkins Q-learning algorithm guarantees asymptotic convergence to optimal values (from which the optimal actions can be derived) provided every state of the MDP has been visited an infinite number of times (Watkins, 1989; Watkins & Dayan, 1992; Jaakkola et al., 1994; Tsitsiklis, 1994). This asymptotic result does not specify a strategy for achieving this infinite exploration, and as such does not provide a solution to the inherent exploitation-exploration trade-off. To address this, Singh et al. (1998) specify two exploration strategies that guarantee both sufficient exploration for asymptotic convergence to optimal actions, and asymptotic exploitation, for both the Q-learning and SARSA algorithms (a variant of Q-learning) (Rummery & Niranjan, 1994; Singh & Sutton, 1996; Sutton, 1995). Gullapalli and Barto (1994) and Jalali and Ferguson (1989) presented algorithms that learn a model of the environment from experience, perform value iteration on the estimated model, and with infinite exploration converge to the optimal policy asymptotically.

These results, and to the best of our knowledge, all other results for reinforcement learning in *general MDPs*, are asymptotic in nature, providing no guarantee on either the number of actions or the computation time the agent requires to achieve near-optimal performance.

On the other hand, non-asymptotic results become available if one considers *restricted* classes of MDPs, if the model of learning is modified from the standard one, or if one changes the criteria for success. Thus, Saul and Singh (1996) provide an algorithm and learning curves (convergence rates) for an interesting special class of MDPs problem designed to highlight a particular exploitation-exploration trade-off. Fiechter (1994; 1997), whose results are closest in spirit to ours, considers only the discounted-payoff case, and makes the learning protocol easier by assuming the availability of a "reset" button that allows his agent to return to a set of start-states at arbitrary times. Others have provided non-asymptotic results for prediction in uncontrolled Markov processes (Schapire & Warmuth, 1994; Singh & Dayan, 1998).

Thus, despite the many interesting previous results in reinforcement learning, the literature has lacked algorithms for learning optimal behavior in *general* MDPs with provably *finite* bounds on the resources (actions and computation time) required, under the standard model of learning in which the agent wanders continuously in the unknown environment. The results presented in this paper fill this void in what is essentially the strongest possible sense.

We present new algorithms for reinforcement learning, and prove that they have *polynomial* bounds on the resources required to achieve near-optimal payoff in *general* MDPs. After observing that the number of actions required to approach the optimal return is lower bounded, for *any* algorithm, by the mixing time T of the optimal policy (in the undiscounted-payoff case) or by the horizon time T (in the discounted-payoff case), we then give algorithms requiring a number of actions and total computation time that are only polynomial in T and the number of states, for both the undiscounted and discounted cases. An interesting aspect of our algorithms is their rather explicit handling of the exploitation-exploration trade-off.

Two important caveats apply to our current results, as well as all the prior results mentioned above. First, we assume that the agent can observe the state of the environment, which may be an impractical assumption for some reinforcement learning problems. Second, we do not address the fact that the state space may be so large that we will have to resort to methods such as function approximation. While some results are available on reinforcement learning and function approximation (Sutton, 1988; Singh et al., 1995; Gordon, 1995; Tsitsiklis & Roy, 1996), and for partially observable MDPs (Chrisman, 1992; Littman et al., 1995; Jaakkola et al., 1995), they are all asymptotic in nature. The extension of our results to such cases is left for future work.

The outline of the paper is as follows: in Section 2, we give standard definitions for MDPs and reinforcement learning. In Section 3, we argue that the mixing time of policies must be taken into consideration in order to obtain finite-time convergence results in the undiscounted case, and make related technical observations and definitions. Section 4 makes similar arguments for the horizon time in the discounted case, and provides a needed technical lemma. The heart of the paper is contained in Section 5, where we state and prove our main results, describe our algorithms in detail, and provide intuitions for the proofs of convergence rates. Section 6 eliminates some technical assumptions that were made for convenience during the main proofs, while Section 7 discusses some extensions of the main theorem that were deferred for the exposition. Finally, in Section 8 we close with a discussion of future work.

2 Preliminaries and Definitions

We begin with the basic definitions for Markov decision processes.

Definition 1 *A* **Markov decision process (MDP)** M *on* **states** $1, \ldots, N$ *and with* **actions** $a_1, \ldots, a_k$*, consists of:*

- *The* **transition probabilities** $P^a_M(ij) \geq 0$*, which for any action a, and any states i and j, specify the probability of reaching state j after executing action a from state i in M. Thus, $\sum_j P^a_M(ij) = 1$ for any state i and action a.*

- *The* **payoff distributions***, for each state i, with mean $R_M(i)$ (where $R_{max} \geq R_M(i) \geq 0$), and variance $Var_M(i) \leq Var_{max}$. These distributions determine the random payoff received when state i is visited.*

For simplicity, we will assume that the number of actions k is a constant; it will be easily verified that if k is a parameter, the resources required by our algorithms scale polynomially with k.

Several comments regarding some benign technical assumptions that we will make on payoffs are in order here. First, it is common to assume that payoffs are actually associated with state-action pairs, rather than with states alone. Our choice of the latter is entirely for technical simplicity, and all of the results of this paper hold for the standard state-action payoffs model as well. Second, we have assumed fixed upper bounds R_{max} and

Var_{max} on the means and variances of the payoff distributions; such a restriction is necessary for finite-time convergence results. Third, we have assumed that expected payoffs are always non-negative for convenience, but this is easily removed by adding a sufficiently large constant to every payoff.

Note that although the actual payoffs experienced are random variables governed by the payoff distributions, for most of the paper we will be able to perform our analyses in terms of the means and variances; the only exception will be in Section 5.5, where we need to translate high expected payoffs into high actual payoffs.

We now move to the standard definition of a stationary and deterministic policy in an MDP.

Definition 2 *Let M be a Markov decision process over states $1, \ldots, N$ and with actions $a_1, \ldots, a_k$. A* **policy** *in M is a mapping $\pi : \{1, \ldots, N\} \rightarrow \{a_1, \ldots, a_k\}$.*

Later we will have occasion to define and use non-stationary policies, that is, policies in which the action chosen from a given state also depends on the time of arrival at that state.

An MDP M, combined with a policy π, yields a standard Markov process on the states, and we will say that π is *ergodic* if the Markov process resulting from π is ergodic (that is, has a well-defined stationary distribution). For the development and exposition, it will be easiest to consider MDPs for which *every* policy is ergodic, the so-called *unichain* MDPs (Puterman, 1994). In a unichain MDP, the stationary distribution of any policy does not depend on the start state. Thus, considering the unichain case simply allows us to discuss the stationary distribution of any policy without cumbersome technical details, and as it turns out, the result for unichains already forces the main technical ideas upon us. Our results generalize to non-unichain (multichain) MDPs with just a small and necessary change to the definition of the best performance we can expect from a learning algorithm. This generalization to multichain MDPs will be given in Section 7. In the meantime, however, it is important to note that the unichain assumption does *not* imply that every policy will eventually visit every state, or even that there exists a single policy that will do so quickly; thus, the exploitation-exploration dilemma remains with us strongly.

The following definitions for finite-length paths in MDPs will be of repeated technical use in the analysis.

Definition 3 *Let M be a Markov decision process, and let π be a policy in*

*M. A T-**path in** M is a sequence p of $T+1$ states (that is, T transitions) of M:*

$$p = i_1, i_2, \ldots, i_T, i_{T+1}. \tag{1}$$

The probability that p is traversed in M upon starting in state i_1 and executing policy π is

$$\mathbf{Pr}^{\pi}_{M}[p] = \Pi_{k=1}^{T} P_M^{\pi(i_k)}(i_k i_{k+1}). \tag{2}$$

We now define the two standard measures of the return of a policy.

Definition 4 *Let M be a Markov decision process, let π be a policy in M, and let p be a T-path in M. The (expected)* **undiscounted return along** *p in M is*

$$U_M(p) = \frac{1}{T}(R_{i_1} + \cdots + R_{i_T}) \tag{3}$$

and the (expected) **discounted return along** *p in M is*

$$V_M(p) = R_{i_1} + \gamma R_{i_2} + \gamma^2 R_{i_3} \cdots + \gamma^{T-1} R_{i_T} \tag{4}$$

where $0 \leq \gamma < 1$ is a **discount factor** *that makes future reward less valuable than immediate reward. The* T-**step** *undiscounted return from state i is*

$$U_M^{\pi}(i,T) = \sum_p \mathbf{Pr}^{\pi}_{M}[p] U_M(p) \tag{5}$$

and the T-**step** *discounted return from state i is*

$$V_M^{\pi}(i,T) = \sum_p \mathbf{Pr}^{\pi}_{M}[p] V_M(p) \tag{6}$$

where in both cases the sum is over all T-paths p in M that start at i. We define $U_M^{\pi}(i) = \lim_{T\to\infty} U_M^{\pi}(i,T)$ and $V_M^{\pi}(i) = \lim_{T\to\infty} V_M^{\pi}(i,T)$. Since we are in the unichain case, $U_M^{\pi}(i)$ is independent of i, and we will simply write U_M^{π}.

Furthermore, we define the **optimal** T-**step undiscounted return** *from i in M by*

$$U_M^{*}(i,T) = \max_{\pi}\{U_M^{\pi}(i,T)\} \tag{7}$$

and similarly, the **optimal** T-**step discounted return** *from i in M by*

$$V_M^{*}(i,T) = \max_{\pi}\{V_M^{\pi}(i,T)\}. \tag{8}$$

Also, $U_M^*(i) = \lim_{T\to\infty} U_M^*(i,T)$ *and* $V_M^*(i) = \lim_{T\to\infty} V_M^*(i,T)$*. Since we are in the unichain case,* $U_M^*(i)$ *is independent of* i*, and we will simply write* U_M^**. The existence of these limits is guaranteed in the unichain case.*

Finally, we denote the maximum possible T*-step return by* G_{max}^T*; in the undiscounted case* $G_{max}^T \leq R_{max}$*, while in the discounted case* $G_{max}^T \leq TR_{max}$*.*

3 The Undiscounted Case and Mixing Times

It is easy to see that if we are seeking results about the undiscounted return of a learning algorithm after a finite number of steps, we need to take into account some notion of the *mixing times* of policies in the MDP. To put it simply, in the undiscounted case, once we move from the asymptotic return to the finite-time return, there may no longer be a well-defined notion of "the" optimal policy. There may be some policies which will eventually yield high return (for instance, by finally reaching some remote, high-payoff state), but take many steps to approach this high return, and other policies which yield lower asymptotic return but higher short-term return. Such policies are simply incomparable, and the best we could hope for is an algorithm that "competes" favorably with *any* policy, in an amount of time that is comparable to the mixing time *of that policy*.

The standard notion of mixing time for a policy π in a Markov decision process M quantifies the smallest number T of steps required to ensure that the distribution on states after T steps of π is within ϵ of the stationary distribution induced by π, where the distance between distributions is measured by the Kullback-Leibler divergence, the variation distance, or some other standard metric. Furthermore, there are well-known methods for bounding this mixing time in terms of the second eigenvalue of the transition matrix P_M^π, and also in terms of underlying structural properties of the transition graph, such as the conductance (Sinclair, 1993). It turns out that we can state our results for a weaker notion of mixing that only requires the expected *return* after T steps to approach the asymptotic return.

Definition 5 *Let* M *be a Markov decision process, and let* π *be an ergodic policy in* M*. The* ϵ**-return mixing time** *of* π *is the smallest* T *such that for all* $T' \geq T$*,* $|U_M^\pi(i,T') - U_M^\pi| \leq \epsilon$ *for all* i*.*

Suppose we are simply told that there is a policy π whose asymptotic return U_M^π exceeds some value R in an unknown MDP M, and that the

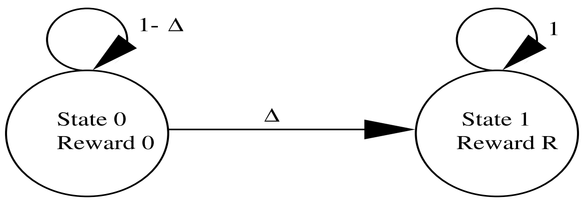

Figure 1: A simple Markov process demonstrating that finite-time convergence results must account for mixing times.

ϵ-return mixing time of π is T. In principle, a sufficiently clever learning algorithm (for instance, one that managed to discover π "quickly") could achieve return close to $U_M^\pi - \epsilon$ in not much more than T steps. Conversely, without further assumptions on M or π, it is not reasonable to expect *any* learning algorithm to approach return U_M^π in many *fewer* than T steps. This is simply because it may take the assumed policy π itself on the order of T steps to approach its asymptotic return. For example, suppose that M has just two states and only one action (see Figure 1): state 0 with payoff 0, self-loop probability $1 - \Delta$, and probability Δ of going to state 1; and absorbing state 1 with payoff $R >> 0$. Then for small ϵ and Δ, the ϵ-return mixing time is on the order of $1/\Delta$; but starting from state 0, it really will require on the order of $1/\Delta$ steps to reach the absorbing state 1 and start approaching the asymptotic return R.

We now relate the notion of ϵ-return mixing time to the standard notion of mixing time.

Lemma 1 *Let M be a Markov decision process on N states, and let π be an ergodic policy in M. Let T be the smallest value such that for all $T' \geq T$, for any state i, the probability of being in state i after T' steps of π is within $\epsilon/(NR_{max})$ of the stationary probability of i under π. Then the ϵ-return mixing time of π is at most $\frac{3TR_{max}}{\epsilon}$.*

The proof of the lemma follows in a straightforward way from the linearity of expectations, and is omitted. The important point is that the ϵ-return mixing time is polynomially bounded by the standard mixing time, but may in some cases be substantially smaller. This would happen, for instance, if the policy quickly settles on a subset of states with common payoff, but takes a long time to settle to its stationary distribution within this subset. Thus, we will choose to state our results for the undiscounted return in terms of the ϵ-return mixing time, but can always translate into the standard notion via Lemma 1.

With the notion of ϵ-return mixing time, we can now be more precise about what type of result is reasonable to expect for the undiscounted case. We would like a learning algorithm such that for *any* T, in a number of actions that is polynomial in T, the return of the learning algorithm is close to that achieved by the *best policy among those that mix in time* T. This motivates the following definition.

Definition 6 *Let M be a Markov decision process. We define $\Pi_M^{T,\epsilon}$ to be the class of all ergodic policies π in M whose ϵ-return mixing time is at most T. We let $opt(\Pi_M^{T,\epsilon})$ denote the optimal expected asymptotic undiscounted return among all policies in $\Pi_M^{T,\epsilon}$.*

Thus, our goal in the undiscounted case will be to compete with the policies in $\Pi_M^{T,\epsilon}$ in time that is polynomial in T, $1/\epsilon$ and N. We will eventually give an algorithm that meets this goal for *every* T and ϵ *simultaneously*. An interesting special case is when $T = T^*$, where T^* is the ϵ-mixing time of the *asymptotically* optimal policy, whose asymptotic return is U^*. Then in time polynomial in T^*, $1/\epsilon$ and N, our algorithm will achieve return exceeding $U^* - \epsilon$ with high probability. It should be clear that, modulo the degree of the polynomial running time, such a result is the best that one could hope for in general MDPs.

4 The Discounted Case and the Horizon Time

For the discounted case, the quantification of which policies a learning algorithm is competing against is more straightforward, since the discounting makes it possible in principle to compete against *all* policies in time proportional to the *horizon time*. In other words, unlike in the undiscounted case, the expected discounted return of *any* policy after $T \approx 1/(1-\gamma)$ steps approaches the expected asymptotic discounted return. This is made precise by the following lemma.

Lemma 2 *Let M be any Markov decision process, and let π be any policy in M. If*

$$T \geq (1/(1-\gamma)) \log(R_{max}/(\epsilon(1-\gamma))) \tag{9}$$

then for any state i,

$$V_M^\pi(i,T) \leq V_M^\pi(i) \leq V_M^\pi(i,T) + \epsilon. \tag{10}$$

We call the value of the lower bound on T given above the ϵ-horizon time *for the discounted MDP M.*

Proof: The lower bound on $V_M^\pi(i)$ follows trivially from the definitions, since all expected payoffs are nonnegative. For the upper bound, fix any infinite path p, and let $R_1, R_2, \ldots$ be the expected payoffs along this path. Then

$$\begin{aligned} V_M(p) &= R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots && (11) \\ &\leq \sum_{k=1}^{T} \gamma^{k-1} R_k + R_{max}\gamma^T \sum_{k=0}^{\infty} \gamma^k && (12) \\ &= V_M(p') + R_{max}\gamma^T(1/(1-\gamma)) && (13) \end{aligned}$$

where p' is the T-path prefix of the infinite path p. Solving

$$R_{max}\gamma^T(1/(1-\gamma)) \leq \epsilon \quad (14)$$

for T yields the desired bound on T; since the inequality holds for every fixed path, it also holds for the distribution over paths induced by any policy π. $\square$

In the discounted case, we must settle for a notion of "competing" that is slightly different than for the undiscounted case. The reason is that while in the undiscounted case, since the total return is always simply *averaged*, a learning algorithm can recover from its "youthful mistakes" (low return during the early part of learning), this is not possible in the discounted case due to the exponentially decaying effects of the discounting factor. The most we can ask for is that, in time polynomial in the ϵ-horizon time, the learning algorithm has a policy that, from its *current state*, has discounted return within ϵ of the asymptotic optimal for that state. Thus, if time were reinitialized to 0, with the current state being the start state, the learned policy would have near-optimal expected return. This is the goal that our algorithm will achieve for general MDPs in the discounted case.

5 Main Theorem

We are now ready to describe our new learning algorithms, and to state and prove our main theorem: namely, that the new algorithms will, for a general MDP, achieve near-optimal performance in polynomial time, where the notions of performance and the parameters of the running time (mixing and horizon times) have been described in the preceding sections. *For ease of exposition only*, we will first state the theorem under the assumption that the learning algorithm is given as input a "targeted" mixing time T, and the optimal return $opt(\Pi_M^{T,\epsilon})$ achieved by any policy mixing within T steps (for the undiscounted case), or the optimal value function $V^*(i)$ (for the discounted case). This simpler case already contains the core ideas of the algorithm and analysis, and these assumptions are entirely removed in Section 6.

Theorem 3 *(Main Theorem) Let M be a Markov decision process over N states.*

- *(Undiscounted case) Recall that $\Pi_M^{T,\epsilon}$ is the class of all ergodic policies whose ϵ-return mixing time is bounded by T, and that $opt(\Pi_M^{T,\epsilon})$ is the optimal asymptotic expected undiscounted return achievable in $\Pi_M^{T,\epsilon}$. There exists an algorithm A, taking inputs ϵ, δ, N, T and $opt(\Pi_M^{T,\epsilon})$, such that the total number of actions and computation time taken by A is polynomial in $1/\epsilon, 1/\delta, N$, T, and R_{max}, and with probability at least $1-\delta$, the total actual return of A exceeds $opt(\Pi_M^{T,\epsilon}) - \epsilon$.*

- *(Discounted case) Let $V^*(i)$ denote the value function for the policy with the optimal expected discounted return in M. Then there exists an algorithm A, taking inputs ϵ, δ, N and $V^*(i)$, such that the total number of actions and computation time taken by A is polynomial in $1/\epsilon, 1/\delta, N$, the horizon time $T = 1/(1-\gamma)$, and R_{max}, and with probability at least $1-\delta$, A will halt in a state i, and output a policy $\hat{\pi}$, such that $V_M^{\hat{\pi}}(i) \geq V^*(i) - \epsilon$.*

The remainder of this section is divided into several subsections, each describing a different and central aspect of the algorithm and proof. The full proof of the theorem is rather technical, but the underlying ideas are quite intuitive, and we sketch them first as an outline.

5.1 High-Level Sketch of the Proof and Algorithms

Although there are some differences between the algorithms and analyses for the undiscounted and discounted cases, for now it will be easiest to think of there being only a single algorithm. This algorithm will be what is commonly referred to as *indirect* or *model-based*: namely, rather than only maintaining a current policy or value function, the algorithm will actually maintain a model for the transition probabilities and the expected payoffs for some *subset* of the states of the unknown MDP M. It is important to emphasize that although the algorithm maintains a partial model of M, it may choose to *never* build a *complete* model of M, if doing so is not necessary to achieve high return.

It is easiest to imagine the algorithm as starting off by doing what we will call *balanced wandering*. By this we mean that the algorithm, upon arriving in a state it has never visited before, takes an arbitrary action from that state; but upon reaching a state it has visited before, it takes the action it has tried the fewest times from that state (breaking ties between actions randomly). At each state it visits, the algorithm maintains the obvious statistics: the average payoff received at that state so far, and for each action, the empirical distribution of next states reached (that is, the estimated transition probabilities).

A crucial notion for both the algorithm and the analysis is that of a *known state*. Intuitively, this is a state that the algorithm has visited "so many" times (and therefore, due to the balanced wandering, has tried each action from that state many times) that the transition probability and expected payoff estimates for that state are "very close" to their true values in M. An important aspect of this definition is that it is weak enough that "so many" times is still polynomially bounded, yet strong enough to meet the simulation requirements we will outline shortly. The fact that the definition of known state achieves this balance is shown in Section 5.2.

States are thus divided into three categories: known states, states that have been visited before, but are still unknown (due to an insufficient number of visits and therefore unreliable statistics), and states that have not even been visited once. An important observation is that we cannot do balanced wandering indefinitely before at least one state becomes known: by the Pigeonhole Principle, we will soon start to accumulate accurate statistics at some state. This fact will be stated more formally in Section 5.5.

Perhaps our most important definition is that of the *known-state MDP*. If S is the set of currently known states, the current known-state MDP is simply an MDP M_S that is naturally *induced* on S by the full MDP M; briefly, all transitions in M *between* states in S are preserved in M_S, while all other transitions in M are "redirected" in M_S to lead to a single additional, absorbing state that intuitively represents all of the unknown and unvisited states.

Although the learning algorithm will not have direct access to M_S, by virtue of the definition of the known states, it will have an *approximation*

$\hat{M}_S$. The first of two central technical lemmas that we prove (Section 5.2) shows that, under the appropriate definition of known state, $\hat{M}_S$ will have good *simulation accuracy*: that is, the expected T-step return of any policy in $\hat{M}_S$ is close to its expected T-step return in M_S. (Here T is either the mixing time that we are competing against, in the undiscounted case, or the horizon time, in the discounted case.) Thus, at any time, $\hat{M}_S$ is a *partial* model of M, for that part of M that the algorithm "knows" very well.

The second central technical lemma (Section 5.3) is perhaps the most enlightening part of the analysis, and is named the "Explore or Exploit" Lemma. It formalizes a rather appealing intuition: either the optimal (T-step) policy achieves its high return by staying, with high probability, in the set S of currently known states — which, most importantly, the algorithm can *detect* and *replicate* by finding a high-return *exploitation* policy in the *partial* model $\hat{M}_S$ — or the optimal policy has significant probability of *leaving* S within T steps — which again the algorithm can detect and replicate by finding an *exploration* policy that quickly reaches the additional absorbing state of the partial model $\hat{M}_S$.

Thus, by performing two off-line, polynomial-time computations on $\hat{M}_S$ (Section 5.4), the algorithm is guaranteed to either find a way to get near-optimal return in M quickly, or to find a way to improve the statistics at an unknown or unvisited state. Again by the Pigeonhole Principle, the latter case cannot occur too many times before a new state becomes known, and thus the algorithm is always making progress. In the worst case, the algorithm will build a model of the entire MDP M, but if that does happen, the analysis guarantees that it will happen in polynomial time.

The following subsections flesh out the intuitions sketched above, providing the full proof of Theorem 3. In Section 6, we show how to remove the assumed knowledge of the optimal return.

5.2 The Simulation Lemma

In this section, we prove the first of two key technical lemmas mentioned in the sketch of Section 5.1: namely, that if one MDP $\hat{M}$ is a sufficiently accurate approximation of another MDP M, then we can actually approximate the T-step return of any policy in M quite accurately by its T-step return in $\hat{M}$.

Eventually, we will appeal to this lemma to show that we can accurately assess the return of policies in the induced known-state MDP M_S by computing their return in the algorithm's approximation $\hat{M}_S$ (that is, we will appeal to Lemma 4 below using the settings $M = M_S$ and $\hat{M} = \hat{M}_S$). The important technical point is that the goodness of approximation required depends only polynomially on $1/T$, and thus the definition of known state will require only a polynomial number of visits to the state.

We begin with the definition of approximation we require.

Definition 7 *Let M and $\hat{M}$ be Markov decision processes over the same state space. Then we say that $\hat{M}$ is an* α-**approximation** *of M if:*

- *For any state i,*

$$R_M(i) - \alpha \leq R_{\hat{M}}(i) \leq R_M(i) + \alpha; \tag{15}$$

- *For any states i and j, and any action a,*

$$P^a_M(ij) - \alpha \leq P^a_{\hat{M}}(ij) \leq P^a_M(ij) + \alpha. \tag{16}$$

We now state and prove the Simulation Lemma, which says that provided $\hat{M}$ is sufficiently close to M in the sense just defined, the T-step return of policies in $\hat{M}$ and M will be similar.

Lemma 4 *(Simulation Lemma) Let M be any Markov decision process over N states.*

- *(Undiscounted Case) Let $\hat{M}$ be an $O((\epsilon/(NTG^T_{max}))^2)$-approximation of M. Then for any policy π in $\Pi^{T,\epsilon/2}_M$ [1], and for any state i,*

$$U^\pi_M(i,T) - \epsilon \leq U^\pi_{\hat{M}}(i,T) \leq U^\pi_M(i,T) + \epsilon. \tag{17}$$

- *(Discounted Case) Let $T \geq (1/(1-\gamma)) \log(R_{max}/(\epsilon(1-\gamma)))$, and let $\hat{M}$ be an $O(\epsilon/(NTG^T_{max}))^2)$-approximation of M. Then for any policy π and any state i,*

$$V^\pi_M(i) - \epsilon \leq V^\pi_{\hat{M}}(i) \leq V^\pi_M(i) + \epsilon. \tag{18}$$

[1] Note that the lemma for the undiscounted case is stated with respect to those policies whose $\epsilon/2$-return mixing time is T, as opposed to ϵ-return mixing time. However, the $\epsilon/2$-return and ϵ-return mixing times are linearly related by standard eigenvalue arguments.

Proof: Let us fix a policy π and a start state i. Let us say that a transition from a state i' to a state j' under action a is β**-small** in M if $P_M^a(i'j') \leq \beta$. Then the probability that T steps from state i following policy π will cross at least one β-small transition is at most βNT. This is because the total probability of all β-small transitions in M from any state i' under action $\pi(i')$ is at most βN, and we have T independent opportunities to cross such a transition. This implies that the total expected contribution to either $U_M^\pi(i,T)$ or $V_M^\pi(i,T)$ by the walks of π that cross at least one β-small transition of M is at most βNTG_{max}^T.

Similarly, since $P_M^a(i'j') \leq \beta$ implies $P_{\hat{M}}^a(i'j') \leq \alpha + \beta$ (since $\hat{M}$ is an α-approximation of M), the total contribution to either $U_{\hat{M}}^\pi(i,T)$ or $V_{\hat{M}}^\pi(i,T)$ by the walks of π that cross at least one β-small transition of M is at most $(\alpha+\beta)NTG_{max}^T$. We can thus bound the difference between $U_M^\pi(i,T)$ and $U_{\hat{M}}^\pi(i,T)$ (or between $V_M^\pi(i,T)$ and $V_{\hat{M}}^\pi(i,T)$) restricted to these walks by $(\alpha+2\beta)NTG_{max}^T$. We will eventually determine a choice for β and solve

$$(\alpha + 2\beta)NTG_{max}^T \leq \epsilon/4 \tag{19}$$

for α.

Thus, for now we restrict our attention to the walks of length T that do not cross any β-small transition of M. Note that for any transition satisfying $P_M^a(i'j') > \beta$, we can convert the additive approximation

$$P_M^a(i'j') - \alpha \leq P_{\hat{M}}^a(i'j') \leq P_M^a(i'j') + \alpha \tag{20}$$

to the multiplicative approximation

$$(1-\Delta)P_M^a(i'j') \leq P_{\hat{M}}^a(i'j') \leq (1+\Delta)P_M^a(i'j') \tag{21}$$

where $\Delta = \alpha/\beta$. Thus, for any T-path p that, under π, does not cross any β-small transitions of M, we have

$$(1-\Delta)^T \mathbf{Pr}_M^\pi[p] \leq \mathbf{Pr}_{\hat{M}}^\pi[p] \leq (1+\Delta)^T \mathbf{Pr}_M^\pi[p]. \tag{22}$$

For any T-path p, the approximation error in the payoffs yields

$$U_M(p) - \alpha \leq U_{\hat{M}}(p) \leq U_M(p) + \alpha \tag{23}$$

and

$$V_M(p) - T\alpha \leq V_{\hat{M}}(p) \leq V_M(p) + T\alpha. \tag{24}$$

Since these inequalities hold for any fixed T-path that does not traverse any β-small transitions in M under π, they also hold when we take expectations over the distributions on such T-paths in M and $\hat{M}$ induced by π. Thus,

$$(1-\Delta)^T[U_M^\pi(i,T)-\alpha]-\epsilon/4 \leq U_{\hat{M}}^\pi(i,T) \leq (1+\Delta)^T[U_M^\pi(i,T)+\alpha]+\epsilon/4 \quad (25)$$

and

$$(1-\Delta)^T[V_M^\pi(i,T)-T\alpha]-\epsilon/4 \leq V_{\hat{M}}^\pi(i,T) \leq (1+\Delta)^T[V_M^\pi(i,T)+T\alpha]+\epsilon/4 \quad (26)$$

where the additive $\epsilon/4$ terms account for the contributions of the T-paths that traverse β-small transitions under π, as bounded by Equation (19).

For the upper bounds, we will use the following Taylor expansion:

$$\log(1+\Delta)^T = T\log(1+\Delta) = T(\Delta - \Delta^2/2 + \Delta^3/3 - \cdots) \geq T\Delta/2. \quad (27)$$

Now to complete the analysis for the undiscounted case, we need two conditions to hold:

$$(1+\Delta)^T U_M^\pi(i,T) \leq U_M^\pi(i,T) + \epsilon/8 \quad (28)$$

and

$$(1+\Delta)^T \alpha \leq \epsilon/8, \quad (29)$$

because then $(1+\Delta)^T[U_M^\pi(i,T)+\alpha]+\epsilon/4 \leq U_M^\pi(i,T)+\epsilon/2$. The first condition would be satisfied if $(1+\Delta)^T \leq 1+\epsilon/8G_{max}^T$; solving for Δ we obtain $T\Delta/2 \leq \epsilon/8G_{max}^T$ or $\Delta \leq \epsilon/(4TG_{max}^T)$. This value of Δ also implies that $(1+\Delta)^T$ is a constant and therefore satisfying the second condition would require that $\alpha = O(\epsilon)$.

Recalling the earlier constraint given by Equation (19), if we choose $\beta = \sqrt{\alpha}$, then we find that

$$(\alpha+2\beta)NTG_{max}^T \leq 3\sqrt{\alpha}NTG_{max}^T \leq \epsilon/4 \quad (30)$$

and

$$\Delta = \alpha/\beta = \sqrt{\alpha} \leq \epsilon/(4TG_{max}^T) \quad (31)$$

and $\alpha = O(\epsilon)$ are all satisfied by the choice of α given in the lemma. A similar argument yields the desired lower bound, which completes the proof for the undiscounted case. The analysis for the discounted case is entirely analogous, except we must additionally appeal to Lemma 2 in order to relate the T-step return to the asymptotic return. □

The Simulation Lemma essentially determines what the definition of known state should be: one that has been visited enough times to ensure (with high probability) that the estimated transition probabilities and the estimated payoff for the state are all within $O((\epsilon/(NTG^T_{max}))^2)$ of their true values. The following lemma, whose proof is a straightforward application of Chernoff bounds, makes the translation between the number of visits to a state and the desired accuracy of the transition probability and payoff estimates.

Lemma 5 *Let M be a Markov decision process. Let i be a state of M that has been visited at least m times, with each action $a_1, \ldots, a_k$ having been executed at least $\lfloor m/k \rfloor$ times from i. Let $\hat{P}^a_M(ij)$ denote the empirical probability transition estimates obtained from the m visits to i. For*

$$m = O(((NTG^T_{max})/\epsilon)^4 \ \ Var_{max} \log(1/\delta)) \tag{32}$$

with probability at least $1 - \delta$, we have

$$|\hat{P}^a_M(ij) - P^a_M(ij)| = O(\epsilon/(NTG^T_{max}))^2) \tag{33}$$

for all states j and actions a, and

$$|\hat{R}_M(i) - R_M(i)| = O(\epsilon/(NTG^T_{max}))^2) \tag{34}$$

where $Var_{max} = \max_i[Var_M(i)]$ is the maximum variance of the random payoffs over all states.

Thus, we get our formal definition of known states:

Definition 8 *Let M be a Markov decision process. We say that a state i of M is* **known** *if each action has been executed from i at least*

$$m_{known} = O(((NTG^T_{max})/\epsilon)^4 \, Var_{max} \log(1/\delta)) \tag{35}$$

times.

5.3 The "Explore or Exploit" Lemma

Lemma 4 indicates the degree of approximation required for sufficient simulation accuracy, and led to the definition of a known state. If we let S denote the set of known states, we now specify the straightforward way in which these known states define an induced MDP. This induced MDP has an additional "new" state, which intuitively represents all of the unknown states and transitions.

Definition 9 *Let M be a Markov decision process, and let S be any subset of the states of M. The* **induced Markov decision process on** S*, denoted M_S, has states $S \cup \{s_0\}$, and transitions and payoffs defined as follows:*

- *For any state $i \in S$, $R_{M_S}(i) = R_M(i)$; all payoffs in M_S are deterministic (zero variance) even if the payoffs in M are stochastic.*
- $R_{M_S}(s_0) = 0$.
- *For any action a, $P^a_{M_S}(s_0 s_0) = 1$. Thus, s_0 is an absorbing state.*
- *For any states $i, j \in S$, and any action a, $P^a_{M_S}(ij) = P^a_M(ij)$. Thus, transitions in M between states in S are preserved in M_S.*
- *For any state $i \in S$ and any action a, $P^a_{M_S}(is_0) = \sum_{j \notin S} P^a_M(ij)$. Thus, all transitions in M that are not between states in S are redirected to s_0 in M_S.*

Definition 9 describes an MDP directly induced on S by the true unknown MDP M, and as such preserves the *true* transition probabilities between states in S. Of course, our algorithm will only have *approximations* to these transition probabilities, leading to the following obvious approximation to M_S: if we simply let $\hat{M}$ denote the obvious empirical approximation to M[2], then $\hat{M}_S$ is the natural approximation to M_S. The following lemma establishes the simulation accuracy of $\hat{M}_S$, and follows immediately from Lemma 4 and Lemma 5.

Lemma 6 *Let M be a Markov decision process, and let S be the set of currently known states of M. Then with probability at least $1 - \delta$,*

- *(Undiscounted Case) For any policy π in $\Pi^{T,\epsilon/2}_{M_S}$, and for any state i,*

$$U^\pi_{M_S}(i, T) - \epsilon \leq U^\pi_{\hat{M}_S}(i, T) \leq U^\pi_{M_S}(i, T) + \epsilon. \tag{36}$$

- *(Discounted Case) Let $T \geq (1/(1-\gamma)) \log(R_{max}/(\epsilon(1-\gamma)))$. Then for any policy π and any state i,*

$$V^\pi_{M_S}(i) - \epsilon \leq V^\pi_{\hat{M}_S}(i) \leq V^\pi_{M_S}(i) + \epsilon. \tag{37}$$

[2] that is, the states of $\hat{M}$ are simply all the states visited so far, the transition probabilities of $\hat{M}$ are the observed transition frequencies, and the rewards are the observed rewards

Let us also observe that any return achievable in M_S (and thus approximately achievable in $\hat{M}_S$) is also achievable in the "real world" M:

Lemma 7 *Let M be a Markov decision process, and let S be the set of currently known states of M. Then for any policy π in M, any state $i \in S$, and any T, $U^{\pi}_{M_S}(i,T) \leq U^{\pi}_{M}(i,T)$ and $V^{\pi}_{M_S}(i,T) \leq V^{\pi}_{M}(i,T)$.*

Proof: Follows immediately from the facts that M_S and M are identical on S, the expected payoffs are non-negative, and that outside of S no payoff is possible in M_S. $\square$

We are now at the heart of the analysis: we have identified a "part" of the unknown MDP M that the algorithm "knows" very well, in the form of the approximation $\hat{M}_S$ to M_S. The key lemma follows, in which we demonstrate the fact that M_S (and thus, by the Simulation Lemma, $\hat{M}_S$) must always provide the algorithm with *either* a policy that will yield large immediate return in the true MDP M, *or* a policy that will allow rapid exploration of an unknown state in M (or both).

Lemma 8 *(Explore or Exploit Lemma) Let M be any Markov decision process, let S be any subset of the states of M, and let M_S be the induced Markov decision process on M. For any $i \in S$, any T, and any $1 > \alpha > 0$, either there exists a policy π in M_S such that $U^{\pi}_{M_S}(i,T) \geq U^{*}_{M}(i,T) - \alpha$ (respectively, $V^{\pi}_{M_S}(i,T) \geq V^{*}_{M}(i,T) - \alpha$), or there exists a policy π in M_S such that the probability that a walk of T steps following π will terminate in s_0 exceeds α/G^{T}_{max}.*

Proof: We give the proof for the undiscounted case; the argument for the discounted case is analogous. Let π be a policy in M satisfying $U^{\pi}_{M}(i,T) = U^{*}_{M}(i,T)$, and suppose that $U^{\pi}_{M_S}(i,T) < U^{*}_{M}(i,T) - \alpha$ (otherwise, π already witnesses the claim of the lemma). We may write

$$U^{\pi}_{M}(i,T) = \sum_{p} \mathbf{Pr}^{\pi}_{M}[p] U_M(p) \qquad (38)$$

$$= \sum_{q} \mathbf{Pr}^{\pi}_{M}[q] U_M(q) + \sum_{r} \mathbf{Pr}^{\pi}_{M}[r] U_M(r) \qquad (39)$$

where the sums are over, respectively, all T-paths p in M that start in state i, all T-paths q in M that start in state i and in which every state in q is in S, and all T-paths r in M that start in state i and in which at least one state is not in S. Keeping this interpretation of the variables p, q and r fixed, we may write

$$\sum_q \mathbf{Pr}^\pi_M[q] U_M(q) = \sum_q \mathbf{Pr}^\pi_{M_S}[q] U_{M_S}(q) \leq U^\pi_{M_S}(i, T). \tag{40}$$

The equality follows from the fact that for any path q in which every state is in S, $\mathbf{Pr}^\pi_M[q] = \mathbf{Pr}^\pi_{M_S}[q]$ and $U_M(q) = U_{M_S}(q)$, and the inequality from the fact that $U^\pi_{M_S}(i, T)$ takes the sum over all T-paths in M_S, not just those that avoid the absorbing state s_0. Thus

$$\sum_q \mathbf{Pr}^\pi_M[q] U_M(q) \leq U^*_M(i, T) - \alpha \tag{41}$$

which implies that

$$\sum_r \mathbf{Pr}^\pi_M[r] U_M(r) \geq \alpha. \tag{42}$$

But

$$\sum_r \mathbf{Pr}^\pi_M[r] U_M(r) \leq G^T_{max} \sum_r \mathbf{Pr}^\pi_M[r] \tag{43}$$

and so

$$\sum_r \mathbf{Pr}^\pi_M[r] \geq \alpha / G^T_{max} \tag{44}$$

as desired. □

5.4 Off-line Optimal Policy Computations

Let us take a moment to review and synthesize. The combination of Lemmas 6, 7 and 8 establishes our basic line of argument:

- At any time, if S is the set of current known states, the T-step return of any policy π in $\hat{M}_S$ (approximately) lower bounds the T-step return of (any extension of) π in M.
- At any time, there must either be a policy in $\hat{M}_S$ whose T-step return in M is nearly optimal, or there must be a policy in $\hat{M}_S$ that will quickly reach the absorbing state — in which case, this same policy, executed in M, will quickly reach a state that is not currently in the known set S.

In this section, we discuss how with two off-line, polynomial-time computations on $\hat{M}_S$, we can find both the policy with highest return (the exploitation policy), and the one with the highest probability of reaching the absorbing state in T steps (the exploration policy). This essentially follows from the fact that the standard value iteration algorithm from the dynamic programming literature is able to find T-step optimal policies for an arbitrary MDP with N states in $O(N^2T)$ computation steps for both the discounted and the undiscounted cases.

For the sake of completeness, we present the undiscounted and discounted value iteration algorithms (Bertsekas & Tsitsiklis, 1989) below. The optimal T-step policy may be non-stationary, and is denoted by a sequence $\pi^* = \{\pi_1^*, \pi_2^*, \pi_3^*, \ldots, \pi_T^*\}$, where $\pi_t^*(i)$ is the optimal action to be taken from state i on the t^{th} step.

T-step Undiscounted Value Iteration:

Initialize: for all $i \in \hat{M}_S$, $U_{T+1}(i) = 0.0$

For $t = T, T-1, T-2, \ldots, 1$:
for all i, $U_t(i) = R_{\hat{M}_S}(i) + \max_a \sum_j P^a_{\hat{M}_S}(ij)U_{t+1}(j)$
for all i $\pi_t^*(i) = argmax_a[R_{\hat{M}_S}(i) + \sum_j P^a_{\hat{M}_S}(ij)U_{t+1}(j)]$

Undiscounted value iteration works backwards in time, first producing the optimal policy for time step T, then the optimal policy for time step $T-1$, and so on. Observe that for finite T a policy that maximizes cumulative T-step return will also maximize the average T-step return.

T-step Discounted Value Iteration:

Initialize: for all $i \in \hat{M}_S$, $V_{T+1}(i) = 0.0$

For $t = T, T-1, T-2, \ldots, 1$:
for all i, $V_t(i) = R_{\hat{M}_S}(i) + \gamma \max_a \sum_j P^a_{\hat{M}_S}(ij)V_{t+1}(j)$
for all i $\pi_t^*(i) = argmax_a[R_{\hat{M}_S}(i) + \gamma \sum_j P^a_{\hat{M}_S}(ij)V_{t+1}(j)]$

Again, discounted value iteration works backwards in time, first producing the optimal policy for time step T, then the optimal policy for time step $T-1$, and so on.

Note that the total computation involved is $O(N^2T)$ for both the discounted and the undiscounted cases.

Our use of value iteration will be straightforward: at certain points in the execution of the algorithm, we will perform value iteration off-line twice: once on $\hat{M}_S$ (using either the undiscounted or discounted version, depending on the measure of return), and a second time on what we will denote $\hat{M}'_S$ (on which the computation will use undiscounted value iteration, regardless of the measure of return).

The MDP $\hat{M}'_S$ has the same transition probabilities as $\hat{M}_S$, but different payoffs: in $\hat{M}'_S$, the absorbing state s_0 has payoff R_{max} and all other states have payoff 0. Thus we reward exploration (as represented by visits to s_0) rather than exploitation. If $\hat{\pi}$ is the policy returned by value iteration on $\hat{M}_S$ and $\hat{\pi}'$ is the policy returned by value iteration on $\hat{M}'_S$, then Lemma 8 guarantees that either the T-step return of $\hat{\pi}$ from our current known state approaches the optimal achievable in M (which for now we are assuming we know, and can thus detect), or the probability that $\hat{\pi}'$ reaches s_0, and thus that the execution of $\hat{\pi}'$ in M reaches an unknown or unvisited state in T steps with significant probability (which we can also detect).

5.5 Putting it All Together

All of the technical pieces we need are now in place, and we now give a more detailed description of the algorithm, and tie up some loose ends. In Section 6, we remove the assumption that we know the optimal returns that can be achieved in M.

In the sequel, we will use the expression *balanced wandering* to denote the steps of the algorithm in which the current state is not a known state, and the algorithm executes the action that has been tried the fewest times before from the current state. Note that once a state becomes known, by definition it is never involved in a step of balanced wandering again. We use m_{known} to denote the number of visits required to a state before it becomes a known state (different for the undiscounted and discounted cases), as given by Definition 8.

We call the algorithm the *Explicit Explore or Exploit* (or E^3) algorithm, because whenever the algorithm is not engaged in balanced wandering, it performs an explicit off-line computation on the partial model in order to find a T-step policy guaranteed to either exploit or explore. In the description that follows, we freely mix the description of the steps of the algorithm with observations that will make the ensuing analysis easier to digest.

The Explicit Explore or Exploit (E^3) Algorithm:

- (Initialization) Initially, the set S of known states is empty.
- (Balanced Wandering) Any time the current state is not in S, the algorithm performs balanced wandering.
- (Discovery of New Known States) Any time a state i has been visited m_{known} times during balanced wandering, it enters the known set S, and no longer participates in balanced wandering.
- **Observation:** Clearly, after $N(m_{known}-1)+1$ steps of balanced wandering, by the Pigeonhole Principle *some* state becomes known. This is the worst case, in terms of the time required for at least one state to become known. More generally, if the total number of steps of balanced wandering the algorithm has performed ever exceeds Nm_{known}, then *every* state of M is known (even if these steps of balanced wandering are not consecutive). This is because each known state can account for at most m_{known} steps of balanced wandering.
- (Off-line Optimizations) Upon reaching a known state $i \in S$ during balanced wandering, the algorithm performs the two off-line optimal policy computations on $\hat{M}_S$ and $\hat{M}'_S$ described in Section 5.4:
  - (Attempted Exploitation) If the resulting exploitation policy $\hat{\pi}$ achieves return from i in $\hat{M}_S$ that is at least $U^* - \epsilon/2$ (respectively, in the discounted case, at least $V^*(i) - \epsilon/2$), the algorithm executes $\hat{\pi}$ for the next T steps (respectively, halts and outputs i and $\hat{\pi}$). Here T is the given $\epsilon/2$-mixing time given to the algorithm as input (respectively, the horizon time).
  - (Attempted Exploration) Otherwise, the algorithm executes the resulting exploration policy $\hat{\pi}'$ (derived from the off-line computation on $\hat{M}'_S$) for T steps in M, which by Lemma 8 is guaranteed to have probability at least $\epsilon/(2G^T_{max})$ of leaving the set S.
- (Balanced Wandering) Any time an attempted exploitation or attempted exploration visits a state not in S, the algorithm immediately resumes balanced wandering.
- **Observation:** Thus, *every action* taken by the algorithm in M is either a step of balanced wandering, or is part of a T-step attempted exploitation or attempted exploration.

This concludes the description of the algorithm; we can now wrap up the analysis.

One of the main remaining issues is our handling of the confidence parameter δ in the statement of the main theorem: for both the undiscounted and discounted case, Theorem 3 ensures that a certain performance guarantee is met with probability at least $1-\delta$. There are essentially three different sources of failure for the algorithm:

- At some known state, the algorithm actually has a poor approximation to the next-state distribution for some action, and thus $\hat{M}_S$ does *not* have sufficiently strong simulation accuracy for M_S.
- Repeated attempted explorations fail to yield enough steps of balanced wandering to result in a new known state.
- (Undiscounted case only) Repeated attempted exploitations fail to result in actual return near U^*.

Our handling of the failure probability δ is to simply allocate $\delta/3$ to each of these sources of failure. The fact that we can make the probability of the first source of failure (a "bad" known state) controllably small is quantified by Lemma 6. Formally, we use $\delta' = \delta/3N$ in Lemma 6 to meet the requirement that all states in M_S be known simultaneously.

For the second source of failure (failed attempted explorations), a standard Chernoff bound analysis suffices: by Lemma 8, each attempted exploration can be viewed as an independent Bernoulli trial with probability at least $\epsilon/(2G^T_{max})$ of "success" (at least one step of balanced wandering). In the worst case, we must make *every* state known before we can exploit, requiring Nm_{known} steps of balanced wandering. The probability of having fewer than Nm_{known} steps of balanced wandering will be smaller than $\delta/3$ if the number of (T-step) attempted explorations is

$$O((G^T_{max}/\epsilon)N\log(N/\delta)m_{known}). \tag{45}$$

We can now finish the analysis for the discounted case. In the discounted case, if we ever discover a policy $\hat{\pi}$ whose return from the current state i in $\hat{M}_S$ is close to $V^*(i)$ (attempted exploitation), then the algorithm is finished by arguments already detailed — since (with high probability) $\hat{M}_S$ is a very accurate approximation of part of M, $\hat{\pi}$ must be a near-optimal policy from i in M as well (Lemma 7). As long as the algorithm is not finished, it must be engaged in balanced wandering or attempted explorations, and we have already bounded the number of such steps before (with high probability) every state is in the known set S. If and when S does contain all states of M, then $\hat{M}_S$ is actually an accurate approximation of the entire MDP M, and then Lemma 8 ensures that exploitation must be possible (since exploration is not). We again emphasize that the case in which S eventually contains all of the states of M is only the worst case for the *analysis* — the algorithm may discover it is able to halt with a near-optimal exploitation policy long before this ever occurs.

Using the value of m_{known} given for the discounted case by Definition 8, the total number of *actions* executed by the algorithm in the discounted case is thus bounded by T times the maximum number of attempted explorations, given by Equation (45), for a bound of

$$O((NTG_{max}^T/\epsilon)\log(N/\delta)m_{known}). \tag{46}$$

The total *computation time* is bounded by $O(N^2T)$ (the time required for the off-line computations) times the maximum number of attempted explorations, giving

$$O((N^3TG_{max}^T/\epsilon)\log(N/\delta)m_{known}). \tag{47}$$

For the undiscounted case, things are slightly more complicated, since we do not want to simply halt upon finding a policy whose expected return is near U^*, but want to achieve *actual* return approaching U^*, which is where the third source of failure (failed attempted exploitations) enters. We have already argued that the total number of T-step attempted explorations the algorithm can perform before S contains all states of M is polynomially bounded. All other actions of the algorithm must be accounted for by T-step attempted exploitations. Each of these T-step attempted exploitations has expected return at least $U^* - \epsilon/2$. The probability that the actual return, *restricted to just these attempted exploitations*, is less than $U^* - 3\epsilon/4$, can be made smaller than $\delta/3$ if the number of blocks exceeds $O((1/\epsilon)^2\log(1/\delta))$; this is again by a standard Chernoff bound analysis. However, we also need to make sure that the return restricted to these exploitation blocks is sufficient to dominate the potentially low return of the attempted explorations. It is not difficult to show that provided the number of attempted exploitations exceeds $O(G_{max}^T/\epsilon)$ times the number of attempted explorations (bounded by Equation (45)), both conditions are satisfied, for a total number of actions bounded by $O(T/\epsilon)$ times the number of attempted explorations, which is

$$O(NT(G_{max}^T/\epsilon)^2\log(N/\delta)m_{known}). \tag{48}$$

The total computation time is thus $O(N^2T/\epsilon)$ times the number of attempted explorations, and thus bounded by

$$O(N^3T(G_{max}^T/\epsilon)^2 \log(N/\delta)m_{known}). \tag{49}$$

This concludes the proof of the main theorem. We remark that no serious attempt to minimize these worst-case bounds has been made; our immediate goal was to simply prove *polynomial* bounds in the most straightforward manner possible. It is likely that a practical implementation based on the algorithmic ideas given here would enjoy performance on natural problems that is considerably better than the current bounds indicate. (See Moore and Atkeson, 1993, for a related heuristic algorithm.)

6 Eliminating Knowledge of the Optimal Returns and the Mixing Time

In order to simplify our presentation of the main theorem, we made the assumption that the learning algorithm was given as input the targeted mixing time T and the optimal return $opt(\Pi_M^{T,\epsilon})$ achievable in this mixing time (in the undiscounted case), or the value function $V^*(i)$ (in the discounted case; the horizon time T is implied by knowledge of the discounting factor γ). In this section, we sketch the straightforward way in which these assumptions can be removed without changing the qualitative nature of the results, and briefly discuss some alternative approaches that may result in a more practical version of the algorithm.

Let us begin by noting that knowledge of the optimal returns $opt(\Pi_M^{T,\epsilon})$ or $V^*(i)$ is used only in the Attempted Exploitation step of the algorithm, where we must compare the return possible from our current state in $\hat{M}_S$ with the best possible in the entire unknown MDP M. In the absence of this knowledge, the Explore or Exploit Lemma (Lemma 8) ensures us that it is safe to have a *bias towards exploration*. More precisely, any time we arrive in a known state i, we will *first* perform the Attempted Exploration off-line computation on the modified known-state MDP $\hat{M}'_S$ described in Section 5.4, to obtain the optimal exploration policy $\hat{\pi}'$. Since it is a simple matter to compute the probability that $\hat{\pi}'$ will reach the absorbing state s_0 of $\hat{M}'_S$ in T steps, we can then compare this probability to the lower bound $\epsilon/(2G_{max}^T)$ of Lemma 8. As long as this lower bound is exceeded, we may execute $\hat{\pi}'$ in an attempt to visit a state not in S. If this lower bound is *not* exceeded, Lemma 8 guarantees that the off-line computation on $\hat{M}_S$ in the

Attempted Exploitation step *must* result in an exploitation policy $\hat{\pi}$ that is close to optimal. As before, in the discounted case we halt and output $\hat{\pi}$, while in the undiscounted case we execute $\hat{\pi}$ in M and continue.

Note that this exploration-biased solution to removing knowledge of $opt(\Pi_M^{T,\epsilon})$ or $V^*(i)$ results in the algorithm always exploring all states of M that can be reached in a reasonable amount of time, before doing any exploitation. Although this is a simple way of removing the knowledge while keeping a polynomial-time algorithm, practical variants of our algorithm might pursue a more balanced strategy, such as the standard approach of having a strong bias towards exploitation instead, but doing enough exploration to ensure rapid convergence to the optimal performance. For instance, we can maintain a *schedule* $\alpha(t) \in [0, 1]$, where t is the total number of actions taken in M by the algorithm so far. Upon reaching a known state, the algorithm performs Attempted Exploitation (execution of $\hat{\pi}$) with probability $1 - \alpha(t)$, and Attempted Exploration (execution of $\hat{\pi}'$) with probability $\alpha(t)$. For choices such as $\alpha(t) = 1/t$, standard analyses ensure that we will still explore enough that $\hat{M}_S$ will, in polynomial time, contain a policy whose return is near the optimal of M, but the return we have enjoyed in the meantime may be much greater than the exploration-biased solution given above. Note that this approach is similar in spirit to the "ϵ-greedy" method of augmenting algorithms such as Q-learning with an exploration component, but with a crucial difference: while in ϵ-greedy exploration, we with probability $\alpha(t)$ attempt a *single* action designed to visit a rarely visited state, here we are proposing that with probability $\alpha(t)$ we execute a *multi-step policy* for reaching an unknown state, a policy that is provably justified by $\hat{M}'_S$.

For the undiscounted case, it still remains to remove our assumption that the algorithm knows the targeted mixing time T. Indeed, we would like to state our main theorem for *any* value of T: that is, for any T, as long as we run the algorithm for a number of steps that is polynomial in T and the other parameters, the total return will exceed $opt(\Pi_M^{T,\epsilon}) - \epsilon$ with high probability. This is easily accomplished: ignoring all other parameters, we already have an algorithm $A(T)$ that, given T as input, runs for $P(T)$ steps for some fixed polynomial $P(\cdot)$ and meets the desired criterion. We now propose a new algorithm A', which does not need T as input, and simply runs A sequentially for $T = 1, 2, 3, \ldots$. For any T, the amount of time A' must be run before A' has executed $A(T)$ is $\sum_{t=1}^{T} P(t) \leq TP(T) = P'(T)$, which is still polynomial in T. We just need to run A' for sufficiently many steps after the first $P'(T)$ steps to dominate any low-return periods that took place in those $P'(T)$ steps, similar to the analysis done for the undiscounted case towards the end of Section 5.5. We again note that this solution, while sufficient for polynomial time, is far from the one we would implement in practice: for instance, we would clearly want to modify the algorithm so that the many sequential executions of A shared and accumulated common partial models of M.

7 The Multichain Case

The main issue in extending our results to arbitrary multichain MDPs is that the asymptotic undiscounted return for any policy π is not independent of the start state. This makes the undiscounted case for multichain MDPs look a lot like the usual discounted case. Indeed, our results extend to arbitrary multichain MDPs in the discounted case without any modification. Therefore, one way to deal with the undiscounted-case multichain MDPs is to only ask that given polynomial time our algorithm will be in a state for which it has a policy that has an expected return that is near-optimal for that state. Another way is to modify what we can expect when we compete against a policy: instead of expecting to compete against the largest asymptotic return over any start state for that policy, we can compete against the lowest asymptotic return over any start state for that policy. Thus, we modify Definitions 5 and 6 as follows:

Definition 5 *Let M be a Markov decision process, and let π be* **any** *policy in M. The* **ϵ-return mixing time** *of π is the smallest T such that for all $T' \geq T$, $|U_M^\pi(i, T') - U_M^\pi(i)| \leq \epsilon$ for all i.*

Definition 6 *Let M be an arbitrary Markov decision process. We define $\Pi_M^{T,\epsilon}$ to be the class of all policies in M whose ϵ-return mixing time is at most T. We let $opt(\Pi_M^{T,\epsilon}) = \max_{\pi \in \Pi_M^{T,\epsilon}} [\min_i U_M^\pi(i)]$, be the optimal expected asymptotic undiscounted return among all policies in $\Pi_M^{T,\epsilon}$.*

Under these refined definitions, all of our undiscounted-case results on unichain MDPs extend without modification to arbitrary MDPs.

8 Future Work

There are a number of interesting lines for further research.

- **Practical implementation.** Although the polynomial bounds proven here are far too large to immediately claim the practical relevance of our algorithm, we feel that the underlying algorithmic ideas are very promising and will eventually result in a competitive algorithm. We are currently examining the practical issues and choices that arise for an implementation, some of which were discussed briefly in Section 6, and we hope to report on an implementation and experiments soon.
- **A model-free version.** Partially related to the last item, it would be nice to find an algorithm similar to ours that did not require maintaining a partial model, but only a policy (or perhaps several). We are currently investigating this as well.
- **Large state spaces.** It would be interesting to study the applicability of recent methods for dealing with large state spaces, such as function approximation, to our algorithm. This has been recently investigated in the context of factored MDPs (Kearns & Koller, 1999).

Acknowledgements

We give warm thanks to Tom Dean, Tom Dietterich, Tommi Jaakkola, Leslie Kaelbling, Michael Littman, Lawrence Saul, Terry Sejnowski, and Rich Sutton for valuable comments. Satinder Singh was supported by NSF grant IIS-9711753 for the portion of this work done while he was at the University of Colorado, Boulder.

References

Barto, A. G., Sutton, R. S., & Watkins, C. (1990). Sequential decision problems and neural networks. In Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems 2*, pages 686–693, San Mateo, CA. Morgan Kaufmann.

Bertsekas, D. P. (1987). *Dynamic Programming: Deterministic and Stochastic Models.* Englewood Cliffs, NJ: Prentice-Hall.

Bertsekas, D. P. & Tsitsiklis, J. N. (1989). *Parallel and Distributed Computation: Numerical Methods.* Englewood Cliffs, NJ: Prentice-Hall.

Bertsekas, D. P. & Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming.* Belmont, MA: Athena Scientific.

Chrisman, L. (1992). Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. In *AAAI-92*.

Fiechter, C. (1994). Efficient reinforcement learning. In *COLT94: Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, pages 88–97. ACM Press.

Fiechter, C. (1997). Expected mistake bound model for on-line reinforcement learning. In *Machine Learning: Proceedings of the Fourteenth International Conference (ICML97)*, pages 116–124. Morgan Kaufmann.

Gordon, G. J. (1995). Stable function approximation in dynamic programming. In Prieditis, A. & Russell, S. (Eds.), *Machine Learning: Proceedings of the Twelth International Conference*, pages 261–268. Morgan Kaufmann.

Gullapalli, V. & Barto, A. G. (1994). Convergence of indirect adaptive asynchronous value iteration algorithms. In Cowan, J. D., Tesauro, G., & Alspector, J. (Eds.), *Advances is Neural Information Processing Systems 6*, pages 695–702. Morgan Kauffman.

Jaakkola, T., Jordan, M. I., & Singh, S. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, *6*(6), 1185–1201.

Jaakkola, T., Singh, S., & Jordan, M. I. (1995). Reinforcement learning algorithm for partially observable Markov decision problems. In Tesauro, G., touretzky, D. S., & Leen, T. K. (Eds.), *Advances in Neural Information Processing Systems 7*, pages 345–352. Morgan Kaufmann.

Jalali, A. & Ferguson, M. (1989). A distributed asynchronous algorithm for expected average cost dynamic programming. In *Proceedings of the 29th Conference on Decision and Control*, pages 1283–1288, Honolulu, Hawaii.

Kearns, M. & Koller, D. (1999). Efficient Reinforcement Learning in Factored MDPs. To appear in IJCAI 1999.

Kumar, P. R. & Varaiya, P. P. (1986). *Stochastic Systems: Estimation, Identification, and Adaptive Control.* Englewood Cliffs, N.J.: Prentice Hall.

Littman, M., Cassandra, A., & Kaelbling., L. (1995). Learning policies for partially observable environments: Scaling up. In Prieditis, A. & Russell, S. (Eds.), *Proceedings of the Twelfth International Conference on Machine Learning*, pages 362–370, San Francisco, CA. Morgan Kaufmann.

Moore, A. W & Atkeson, C. G.. (1993). Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning*, *12*(1).

Puterman, M. L. (1994). *Markov decision processes : discrete stochastic dynamic programming*. New York: John Wiley & Sons.

Rummery, G. A. & Niranjan, M. (1994). On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University Engineering Dept.

Saul, L. & Singh, S. (1996). Learning curve bounds for markov decision processes with undiscounted rewards. In *COLT96: Proceedings of the Ninth Annual ACM Conference on Computational Learning Theory*.

Schapire, R. E. & Warmuth, M. K. (1994). On the worst-case analysis of temporal-difference learning algorithms. In Cohen, W. W. & Hirsh, H. (Eds.), *Machine Learning: Proceedings of the Eleventh International Conference*, pages 266–274. Morgan Kaufmann.

Sinclair, A. (1993). *Algorithms for random generation and counting: a Markov chain approach*. Boston: Birkhauser.

Singh, S. & Dayan, P. (1998). Analytical mean squared error curves for temporal difference learning. *Machine Learning*. in press.

Singh, S., Jaakkola, T., & Jordan, M. I. (1995). Reinforcement learning with soft state aggregation. In *Advances in Neural Information Processing Systems 7*. Morgan Kaufmann.

Singh, S., Jaakkola, T., Littman, M. L., & Szepesvari, C. (1998). Convergence results for single-step on-policy reinforcement learning algorithms. *Machine Learning*. in press.

Singh, S. & Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. *Machine Learning*, *22*, 123–158.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*, 9–44.

Sutton, R. S. (1995). Generalization in reinforcement learning: Successful examples using sparse coarse coding. In Touretzky, D. S., Mozer, M. C., & Hasselmo, M. E. (Eds.), *Advances in Neural Information Processing Systems 8*, pages 1038–1044. MIT Press.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction.* Cambridge, MA: MIT Press.

Thrun, S. B. (1992). The role of exploration in learning control. In White, D. A. & Sofge, D. A. (Eds.), *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches.* Florence, Kentucky 41022: Van Nostrand Reinhold.

Tsitsiklis, J. (1994). Asynchronous stochastic approximation and Q-learning. *Machine Learning*, *16*(3), 185–202.

Tsitsiklis, J. & Roy, B. V. (1996). Feature-based methods for large scale dynamic programming. *Machine Learning*, *22*, 59–94.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards.* PhD thesis, Cambridge Univ., Cambridge, England.

Watkins, C. J. C. H. & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3/4), 279–292.