Near-Optimal Reinforcement Learning in Polynomial Time

Michael Kearns*	Satinder Singh
Syntek Capital	Syntek Capital
New York, NY 10019	New York, NY 10019
mkearns@cis.upenn.edu	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** $P_M^a(ij) \geq 0$*, 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,* $\sum_j P_M^a(ij) = 1$ *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*

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

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

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

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

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

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

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

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

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

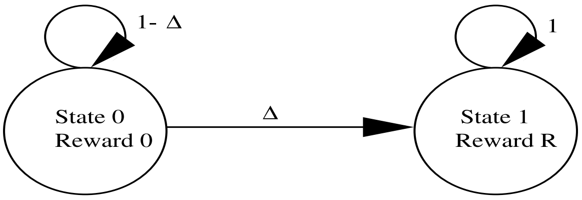

Figure 1: A simple Markov process demonstrating that finite-time convergence results must account for mixing times.

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

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

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. □

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 tec