Exploration in Metric State Spaces

Sham Kakade **Michael Kearns** **John Langford**
Gatsby Unit Department of Computer and Information Science
University College London University of Pennsylvania
London, England Philadelphia, Pennsylvania
sham@gatsby.ucl.ac.uk,mkearns@cis.upenn.edu,jcl+@cs.cmu.edu

Abstract

We present a provably near-optimal algorithm for reinforcement learning in Markov decision processes in which there is a natural *metric* on the state space that allows the construction of accurate local models. Our algorithm is a generalization of the E^3 algorithm of Kearns and Singh, and assumes a black box for approximate planning. Unlike the original E^3, our algorithm finds a near optimal policy in an amount of time that does not directly depend on the size of the state space, but instead depends on the *covering* numbers of the state space, which are informally the number of neighborhoods in the state space required for accurate local modeling at a chosen resolution.

1 Introduction, Motivation, and Background

Recent years have seen the introduction and study of a number of representational approaches to Markov Decision Processes (MDPs) with very large or infinite state spaces. These include the broad family known as function approximation, in which a parametric functional form is used to approximate value functions, and direct models of the underlying dynamics and rewards, such as factored or Dynamic Bayes Net (DBN) MDPs. For each of these approaches, there are now at least plausible heuristics, and sometimes formal analyses, for problems of planning and learning.

Less studied and more elusive has been the problem of *global exploration*, or managing the exploration-exploitation trade-off. Here the goal is to learn a (globally) near-optimal T-step policy in an amount of time that has no direct dependence on the state space size, but only on the complexity of the chosen representation. Indeed, it is only recently that provably correct and efficient algorithms for exploration in small state spaces became known (such as the E^3 algorithm[3] and its generalizations[4]). This approach has been generalized to factored MDPs under certain assumptions [2], but there remain many unresolved questions regarding efficient exploration in large MDPs, including whether model-based approaches are required [1].

In this work, we examine the problem of exploration in environments in which there is a *metric* on state-action pairs with the property that "nearby" state-actions have "similar"

[1] We note that the recent work on gradient methods for approximate planning ([9, 1]) does not address exploration in the strong sense we are interested in here, but instead examines convergence to policies which small amounts of *random* exploration cannot improve (local optimality). In general, effective exploration may require the careful *planning* of a long sequence of steps that will never be encountered by a random walk. See [6] for a further discussion.

dynamics. Such conditions are common for navigation-like problems, but may be more broadly applicable as well. Furthermore, in such metric environments, it may often be plausible to assume that (a) sufficiently dense experience in a neighborhood of state space allows the construction of approximate models for the local dynamics, and (b) models of local dynamics can be "pieced together" and used for planning on a subset of the global state space.

We formalize these natural and general assumptions, and prove that they are sufficient for global exploration in an amount of time depending on the metric resolution, but *not* on the size of the state space. More precisely, we give a generalization of the E^3 algorithm for metric MDPs which learns a (globally) approximately optimal T-step policy in time depending only on the *covering numbers*, a natural and standard notion of the resolution required for local modeling under the metric. This algorithm succeeds in a model in which learning must take place from a single, unbroken chain of experience — no "resets" to previously visited states are provided.

Metric MDPs are a natural complement to more direct parametric assumptions on value functions and dynamics, and are closely related to a variety of state aggregation methods[8]. Our results provide evidence that, as for factored environments[2], effective exploration mechanisms may be available for metric MDPs.

2 Definitions and Assumptions

We work in the standard MDP setting. Let $P(s'|a,s)$ be the probability of a state s' given an action a and state s. Let $R(s)$ be the reward received in state s. For simplicity, we assume that all rewards are deterministic and fall in the interval $[0,1]$. Define $V_M(\pi,s) \equiv E_{(s_1,s_2,\ldots s_T)\sim\pi,s,M}\frac{1}{T}\sum_{t=1}^{T}R(s_t)$ to be the average reward received over T steps starting from state s while acting under π in MDP M.

We first formalize the assumption that there is a notion of distance that permits local modeling of dynamics. Thus, let $d((s',a'),(s,a))$ measure the "distance" between two state-action pairs. Our results will require that this metric obey $d((s,a),(s,a))=0$ for all (s,a), and symmetry (*i.e.,* $d((s,a),(s',a'))=d((s',a'),(s,a))$ for all $(s,a),(s',a')$), but they do *not* require the triangle inequality. This is fortunate since demanding the triangle inequality limits the applicability of the notion in several natural scenarios. We use t_{metric} to denote the time required to evaluate the metric.

We now provide a standard definition of coverings under a metric. An α*-cover* is a set C of state-action pairs with the property that for any (s,a), there exists $(s',a')\in C$ such that $d((s,a),(s',a'))\leq\alpha$. We define $N(\alpha)$ to be the size of the *largest minimal* α-cover — that is, the largest α-cover C such that the removal of any (s,a) would render C no longer a cover.

Our first assumption is that the metric permits local modeling of dynamics:

Local Modeling Assumption. There exists an algorithm *Model* such that, for any (s,a), if we give *Model* any m samples of the form $(s',a',s'',R(s''))$ in which all $d((s,a),(s',a'))\leq\alpha$, and s'' is distributed according to $P(s''|s',a')$, then *Model* outputs a state $\hat{s}$ and a reward $\hat{R}$, where $\hat{s}$ is drawn according to a $\hat{P}(\hat{s}|s,a)$ satisfying $\sum_{\hat{s}}|\hat{P}(\hat{s}|s,a)-P(\hat{s}|s,a)|\leq\alpha$, and $|R(\hat{s})-\hat{R}|\leq\alpha$. We assume that *Model* runs in time t_{model}.

Thus, with a sufficient number m of local state-action experiences, *Model* can form an accurate approximation of the local environment. Note that there is *no* requirement that a *destination* state $\hat{s}$ be in the neighborhood of (s,a) — we ask only that nearby state-actions permit generalization in next-state distributions, not that these distributions be on nearby states. We provide natural examples in the next subsection where our Local Modeling Assumption can be met, but we expect there are many rather different ones as well.

In addition to an assumption about our ability to build local (generative) models, we must make an assumption about our ability to use such models in planning.

Approximate Planning Assumption. There exists an algorithm, *Plan*, which given a generative model for an unknown MDP M and a state s, returns a policy π whose average reward $V_M(\pi, s)$ satisfies $V_M(\pi, s) > V_M(\pi^*, s) - \beta$, where π^* is the optimal T-step policy from s. We assume that *Plan* runs in time t_{plan} with at most c_{gen} calls to the generative model.

Note that the Local Modeling Assumption does not reduce the state space size, so for an arbitrary and large MDP, great computational resources may be required to meet the Approximate Planning Assumption. Our purpose is not to falsely diminish the difficulty of this task, but to abstract it away from the problem of exploration-exploitation.

There are at least three broad scenarios where this assumption might be met. The first is settings where specialized planning heuristics can do approximate planning due to strong parametric constraints on the state dynamics. For example, the recent work on planning heuristics for factored MDPs is of this form. The second is the *sparse sampling* [5] approach, in which it has been shown that the Approximate Planning Assumption can in fact be met for arbitrary finite-action MDPs by a policy that uses a generative model as a subroutine. Here the sample complexity c_{gen} is exponential in T per state visited (see [5]), but has *no* dependence on the state space size. The third setting requires a regression algorithm that is capable of accurately estimating the value of a given policy. This algorithm can be used iteratively to find a near-optimal policy [6].

At a high level, then, we have introduced the notion of a metric over state-actions, an assumption that this metric permits the construction or inference of local models, and an assumption that such models permit planning. We believe these assumptions are broadly consistent with many of the current proposals on large state spaces. We now provide an example that demonstrates the role of covering numbers, and then we show that our assumptions are sufficient for solving the exploration-exploitation problem in time depending not on the size of the state space, but on the (hopefully much smaller) covering numbers under the metric.

2.1 An Example

We can imagine at least two natural scenarios in which the Local Modeling Assumption might be met. One of these is where there is sufficient sensor information and advance knowledge of the expected effects of actions that the local modeling assumption can be satisfied even with $m = 1$ above. As a simple example, people can typically predict the approximate effects of most physical actions available to them immediately upon entering a room and seeing its layout and content (e.g., if I go left I will exit through that door; if I go straight I will hit that wall). They could not make such predictions for unfamiliar distant rooms. Consider the MDP where the state space is the Euclidean maze world shown in Figure 1.(a), and where the agent is equipped with a vision sensor. In this world, it is plausible that the local dynamics can be predicted at any “seen” location. To apply our analysis, we must first specify a metric. The obvious choice is $d_{\text{sight}}((s, a), (s', a')) = 0$ if there exists line-of-sight between s and s' and ∞ otherwise. Note that this metric satisfies symmetry, but not the triangle inequality (which would be somewhat unnatural in this setting). For any $\alpha \geq 0$, the covering number $N(\alpha)$ is the maximum number of points which can be positioned in the space so that no pair have line-of-sight. One maximal set is given by the dots in Figure 1.(b). Note that even though this a continuous state space, the covering number is much smaller, and naturally determined by the geometric properties of the domain.

If it is unrealistic to assume that we can model the local dynamics at distant locations as well

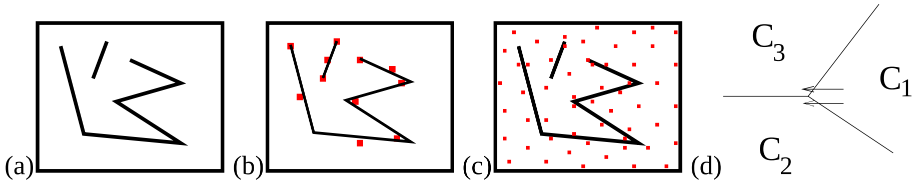

Figure 1: (a) a maze world (b) a largest minimal cover for the line-of-sight metric (c) a largest minimal cover for the line of sight + Euclidean distance metric. (d) an example showing how simple state space aggregation does not work.

as we can at near locations, we can allow our modeling error to grow with distance (e.g., due to lighting conditions). In this case, a reasonable alternative is to define $d((s,a),(s',a')) = d_{\text{sight}}((s,a),(s',a')) + cd_{\text{euclidean}}((s,a),(s',a'))$ where c is a constant controlling the rate of modeling error with Euclidean distance. Using this metric, the covers shown in Figure 1.(c) might naturally arise. Note that our metric is (in general) sensitive to actions as well as states, allowing more flexibility than we have used here.

The above examples are applicable to the $m = 1$ case of the Local Modeling Assumption. The second natural case is the more general "learning" setting, in which the next-state dynamics permit some parameterization that is smooth with respect to the distance metric, thus allowing a finite sample of an environment to provide enough data to fit a parametric next-state distribution for the neighborhood. For instance, if reward appeared stochastically in some region, it might be necessary to visit nearby states a number of times before this distribution is learned. Alternatively, the dynamics could be different in different parts of the state space. For instance, a skier moving down a hill has dynamics dependent on the terrain conditions, such as slope, snow type, and other factors.

Incidentally, Figure 1.(d) illustrates the reason why standard state space aggregation techniques [8] do not work here. In particular, for partitioning induced by a cover on a Euclidean spaces there exist "corners" where 3 (or more) sets meet. When taking actions "toward" this corner from within one of the sets, the distribution over the next aggregate state set is inherently unstable.

3 Metric-E^3

Our algorithm, Metric-E^3, is a direct generalization of the E^3 algorithm[3]. We first outline this original algorithm. A crucial notion in E^3 is that of a "known" state — a state visited often enough such that the dynamics and rewards are accurately modeled at this state. When the agent is not in the current set of known states, the agent wanders randomly to obtain new information. While at a known state, it must decide whether to explore or exploit — a decision which can be made efficiently. Intuitively, the decision to explore is made by determining how much potential reward the agent can obtain by "escaping" the known states to get maximal reward elsewhere. If this number is sufficiently large, the agent explores. This number can be computed by *planning* to "escape" in a fictitious MDP M_{explore} which provides maximal reward for entering an unknown state. The crucial step in the proof of E^3 is showing that either the agent exploits for near optimal reward, or it can explore *quickly*, which results in increasing the size of the set of known states. Since the size of the known set is bounded, the algorithm eventually exploits and obtains near optimal reward.

Metric E^3 has a few key differences. Here, a "known" state-action is a pair (s,a) meeting the antecedent of the Local Modeling Assumption — namely, any pair (s,a) for which the algorithm has obtained at least m α-close experiences $(s',a',s'',R(s''))$. Unlike in E^3, our algorithm does not explicitly enumerate this set of known states, but rather is only able to decide if a particular state-action is known. Thus, in the most general version of our algorithm, our model of the MDP is represented simply by a list of all prior experience.

As in the original E^3, a key step in Metric-E^3 is the creation of the *known* MDP — a model for just that part of the global MDP that we can approximate well. Here the known MDP at any moment is given as a generative model that "patches together" in a particular way the generative models provided by the planning algorithm at known states. More precisely, the *approximate known MDP generative model* takes any state-action (s, a) and a flag bit *exploit* and operates as follows:

1. If (s, a) is not a known state-action, output "fail" and halt.
2. Else give (s, a) and the m prior experiences $(s', a', s'', R(s''))$ in the ϵ-neighborhood of (s, a) to algorithm *Model*; let the resulting outputs be $\hat{s}$ and $\hat{r}$.
3. If *exploit* is 1, set $r \leftarrow \hat{r}$ and $q \leftarrow 0$; otherwise $r \leftarrow 0$ and $q \leftarrow 1$.
4. If for some action $\hat{a}$, the pair $(\hat{s}, \hat{a})$ is itself a known state-action, output $\hat{s}$ and r and halt.
5. Else output a special state z and reward q and halt.

Intuitively, we have described a generative model for two MDPs with identical transition dynamics, but differing rewards according to the value of the *exploit* bit. In both models, all transitions that end in a state with no known actions are "redirected" to a single, special absorbing state z, while all other transitions of the global MDP are preserved. Thus initially the known MDP dynamics are a small subset of the global MDP, but over time may cover much or all of the global state space. For rewards, when *exploit* is 1, rewards from the real environment are preserved, whereas when *exploit* is 0, reward is obtained only at the absorbing state, thus rewarding (rapid) exploration (escape from known state-actions). We shall use $\hat{M}_{\text{exploit}}$ to denote the MDP corresponding to the generative model above when the *exploit* input bit is set to 1, and $\hat{M}_{\text{explore}}$ to denote the MDP generated by setting *exploit* to 0.

Note that under our assumptions, we can always simulate the approximate known MDP generative model. We can also view it as being an approximate (hence the name) generative model for what we shall call the *true known MDP* — the MDP whose generative model is exactly the same as described above, but in which we assume that the local modeling algorithm *Model* is perfect (that is, in the Local Modeling Assumption, $d((s, a), (s', a')) \leq \alpha$ implies $\sum_{\hat{s}} |\hat{P}(\hat{s}|s, a) - P(\hat{s}|s, a)| = 0$, and $|R(\hat{s}) - \hat{R}| = 0$). This may still be only a partial model of the global MDP, but it has the true probabilities for all known state-actions. We shall use M_{exploit} to denote the MDP corresponding to the generative model above with a perfect *Model* and the *exploit* input bit set to 1, and M_{explore} to denote the MDP generated with a perfect *Model* and *exploit* set to 0.

Now we outline the full Metric-E^3 algorithm. It is important to emphasize that this algorithm *never* needs to explicitly enumerate the set of known state-actions.

Algorithm Metric-E^3
Input: $d(\cdot, \cdot)$***, Model, Plan***
Output: A policy π

1. Use random moves until encountering a state s with at least one known action a (that is, where there are at least m α-close previous experiences to (s, a)).
2. Execute *Plan* twice, once using the generative model for $\hat{M}_{\text{exploit}}$ and once using the generative model for $\hat{M}_{\text{explore}}$. Let the resulting policies be π_{exploit} and π_{explore}, respectively.
3. If $V_{\hat{M}_{\text{explore}}}(\pi_{\text{explore}}, s) > \epsilon$, execute π_{explore} for the next T steps, then go to Step 1.

4. Else, HALT and output π_{exploit}.

The claim is that this algorithm finds a near optimal policy, in sample complexity and running time that depend only on the covering number under the metric. We now turn to the analysis.

4 Metric-E^3 Analysis

We first state the main theorems of the paper.

Theorem 4.1 *Let π^* be an optimal policy in M. With probability $1-\delta$, after at most $\frac{TmN(\alpha)}{\epsilon-\alpha(T+1)}\ln(1/\delta)+mN(\alpha)$ actions in M, Metric-E^3 halts in a state s, and output a policy π such that $V_M(\pi,s) \geq V_M(\pi^*,s)-\epsilon-2\beta-2\alpha(T+1)$. Here T is the time horizon, m and α are defined in the Local Modeling Assumption, β is defined in the Approximate Planning Assumption, ϵ is an accuracy parameter, and δ a confidence parameter.*

This shows that the sample complexity (the number of actions required) is bounded in terms of the covering number $N(\alpha)$ (and not the size of the state space). In addition to bounding the sample complexity, we bound the time complexity.

Theorem 4.2 *Let k be the overall sample complexity. Metric-E^3 runs in time at most $\frac{k(k-1)}{2}t_{metric}+2k(t_{plan}+c_{gen}t_{model})+O(k)$.*

A few lemmas are useful in the proofs. First we define $\hat{M}$ to be an α-approximation of M if for all states s, $\sum_{s'}|\hat{P}(s'|s,a)-P(s'|s,a)| \leq \alpha$, and $|R(s)-\hat{R}(s)| \leq \alpha$. The original Simulation Lemma for E^3 had a dependence on the size of the state space that we cannot tolerate in our setting, so we first need an improved version:

Lemma 4.3 *(Simulation Lemma) If $\hat{M}$ is an α-approximation of M, then for any initial state s, any horizon T, and any policy π,*

$$|V_{\hat{M}}(\pi,s)-V_M(\pi,s)| \leq \alpha(T+1)$$

Proof. Let $H_t=\{(s_1,s_2,\ldots s_t)\}$ be the set of length t paths. For $h \in H_t$, let h_t be the t-th state in h and let $A_t(h)$ and $\hat{A}_t(h)$ be the probability of h in M and $\hat{M}$, respectively. Let $Q(s'|s)$ and $\hat{Q}(s'|s)$ be the transition probabilities under π in M and $\hat{M}$, respectively. Since $\hat{M}$ is an α-approximation, for any state s', $\sum_s|Q(s|s')-\hat{Q}(s|s')| \leq \alpha$. Then

$$\begin{aligned}
& \sum_{h\in H_{t+1}} |A_{t+1}(h)-\hat{A}_{t+1}(h)| \\
= & \sum_{h\in H_t,s} |A_t(h)Q(s|h_t)-\hat{A}_t(h)\hat{Q}(s|h_t)| \\
\leq & \sum_{h\in H_t,s} |A_t(h)Q(s|h_t)-\hat{A}_t(h)Q(s|h_t)|+|\hat{A}_t(h)Q(s|h_t)-\hat{A}_t(h)\hat{Q}(s|h_t)| \\
= & \sum_{h\in H_t,s} Q(s|h_t)|A_t(h)-\hat{A}_t(h)|+\hat{A}_t(h)|Q(s|h_t)-\hat{Q}(s|h_t)| \\
\leq & \sum_{h\in H_t} |A_t(h)-\hat{A}_t(h)|+\alpha
\end{aligned}$$

where we have used the triangle inequality and linearity of expectation. Induction on t implies that:

$$\sum_{\text{length } T \text{ paths}} \left| \Pr_{s_i\sim M,\pi,s_0}(s_1,\ldots,s_T) - \Pr_{s_i\sim\hat{M},\pi,s_0}(s_1,\ldots,s_T)\right| \leq \alpha T.$$

Since the rewards $\hat{R}$ in $\hat{M}$ are also α-accurate,

$$\left| V_{\hat{M}}(\pi, s) - E_{\text{length } T \text{ paths in } \hat{M}} \left[\frac{1}{T} \sum_{t=1}^{T} R(s_t) \right] \right| \leq \alpha \, .$$

The result follows using the previous two equations. ■

Now we restate the "Explore-or-Exploit" lemma from [3].

Lemma 4.4 *(Explore or Exploit) Let π^* be the optimal policy for the global MDP M, and let π^*_{exploit} be the optimal policy for the true known MDP M_{exploit} described above. Then for any state s of M_{exploit} and for any $0 < \epsilon < 1$, either*

$$V_{M_{\mathit{exploit}}}(\pi^*_{\mathit{exploit}}, s) > V_M(\pi^*, s) - \epsilon$$

or the optimal policy π^*_{explore} for M_{explore} has probability of at least ϵ of leaving the known states in T steps in M.

One subtle distinction from the original E^3 algorithm exists. Here, although the algorithm plans to reach some unknown state, by the time this state is reached, it might actually be known due to the Local Modeling Assumption. Note that in the maze world example, the agent might plan to escape by moving around a corner. However, when actually executing this escape policy, the states around the corner could become known before they are reached in T steps, if they come into line of sight beforehand.

We now establish that Metric-E^3 ceases to explore in a reasonable amount of time. In the original E^3 this was a consequence of the Pigeonhole Principle applied to the number of states. A similar statement holds here, but now we use the size of a cover under the metric. It is important to note that this lemma holds whether or not the covering number $N(\alpha)$ is known.

Lemma 4.5 *(Exploration Bound) Metric-E^3 encounters at most $mN(\alpha)$ unknown state-actions.*

Proof. First, consider the $m = 1$ case. We construct a set C as follows: the state-action (s, a) at time t is added to the set C if

$$\forall (s', a') \in C: \;\; d((s, a), (s', a')) > \alpha \, .$$

Note that the state at time t is unknown if and only if

$$\forall (s', a') \in \{\text{state-actions visited before time } t\}: \;\; d((s, a), (s', a')) > \alpha$$

and so if (s, a) is unknown, then it is added to C. Thus, the size of C at time t is an upper bound on the number of unknown state-action pairs encountered by the algorithm before time t. Since no element of C covers another element in C, C is minimal. In particular, if any element is removed from C the set of states covered by C will be reduced. It follows that for all t the size of C is less than $N(\alpha)$, and hence the algorithm cannot encounter more than $N(\alpha)$ unknown state-actions.

For the general m case, consider constructing m different sets, $C_1, \ldots, C_m$. The state action at time t is added to only one of the sets C_i if there is no α-close element in C_i. By an analogous argument, if a state-action is unknown, it is added to some C_i, and so the sum of sizes of C_i bounds the number of unknown state-actions encountered by the algorithm before time t. Again, by construction, each C_i is minimal for all t. Hence, the size of each C_i is bounded by $N(\alpha)$ and so the number of unknown state-actions encountered by the algorithm is bounded by $mN(\alpha)$. ■

We now provide the proofs of the main theorems.

Proof of 4.1. The exploration bound of Lemma 4.5 implies we encounter a known state after a number of actions that is at most $mN(\alpha)$, which bounds the number of successful exploration attempts. Each attempted exploration occurs when $V_{\hat{M}_{\text{explore}}}(\pi_{\text{explore}}, s) > \epsilon$, and so $V_{M_{\text{explore}}}(\pi_{\text{explore}}, s) > \epsilon - \alpha(T+1)$. By definition of M_{explore}, the chance of successful exploration is greater than $\epsilon - \alpha(T+1)$. Hence, at most, $\frac{TmN(\alpha)}{\epsilon-\alpha(T+1)} \ln(1/\delta)$ actions successful exploration of the state spaces occurs with a δ chance of error. The total number of actions before halting is less than the sum of the exploration actions known states and the actions taken in unknown states.

The decision to halt occurs when $V_{\hat{M}_{\text{explore}}}(\pi_{\text{explore}}, s) \leq \epsilon$, which implies $V_{M_{\text{explore}}}(\pi^*_{\text{explore}}, s) \leq \epsilon + \alpha(T+1) + \beta$ due to planning and simulation error. By the Explore or Exploit lemma

$$V_{M_{\text{exploit}}}(\pi^*_{\text{exploit}}, s) > V_M(\pi^*, s) - \epsilon - \alpha(T+1) - \beta\ .$$

Due to simulation and planning error in computing an optimal policy in M_{exploit},

$$V_{M_{\text{exploit}}}(\pi_{\text{exploit}}, s) > V_M(\pi^*, s) - \epsilon - 2\alpha(T+1) - 2\beta\ .$$

The result follows since a policy in M has no less reward than in M_{exploit}. ■

Proof of 4.2. It is never necessary to evaluate the metric between two samples more than once. There are at most $\frac{k(k-1)}{2}$ pairs of samples, so line 1 of Metric-E^3 take time at most $t_{\text{metric}}\frac{k(k-1)}{2}$ computation. Step 2 is executed at most k times since at least one transition occurs before reentering step 2. One call to *Plan* requires time at most $t_{\text{plan}} + c_{\text{gen}} t_{\text{model}}$ so the total time spent on step 2 is at most $2k(t_{\text{plan}} + c_{\text{gen}} t_{\text{model}})$. Step 3 takes total time at most $O(k)$. The result follows by adding these times. ■

References

[1] J. Baxter and P. L. Bartlett. “Direct Gradient-Based Reinforcement Learning: I. Gradient Estimation Algorithms”. Technical report , Australian National University, 1999.

[2] M. Kearns and D. Koller. “Efficient Reinforcement Learning in Factored MDPs”. Proceedings of IJCAI, 1999.

[3] M. Kearns and S. Singh. “Near-Optimal Reinforcement Learning in Polynomial Time”. Proceedings of ICML, 1998.

[4] R. I. Brafman and M. Tennenholtz. “R-max – A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning”. Proceedings of IJCAI, 2001.

[5] M. Kearns, Y. Mansour, and A. Ng. “A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes”. Proceedings of IJCAI, 1999.

[6] S. Kakade and J. Langford. “Approximately Optimal Approximate Reinforcement Learning”. Proceedings of ICML, 2002.

[7] A. W. Moore. “The Parti-game Algorithm for Variable Resolution Reinforcement Learning in Multidimensional State-spaces”. In NIPS 6, 1993.

[8] T. Dean and R. Given. “Model Minimization in Markov Decision Processes”. In AAAI, 1997.

[9] R. Sutton, D. McAllester, S. Singh and Y. Mansour. “Policy Gradient Methods for Reinforcement Learning with Function Approximation”. In NIPS 13, 2000.