Provably Safe PAC-MDP Exploration Using Analogies

Melrose Roderick **Vaishnavh Nagarajan** **J. Zico Kolter**
Carnegie-Mellon University

Abstract

A key challenge in applying reinforcement learning to safety-critical domains is understanding how to balance exploration (needed to attain good performance on the task) with safety (needed to avoid catastrophic failure). Although a growing line of work in reinforcement learning has investigated this area of "safe exploration," most existing techniques either 1) do not guarantee safety during the actual exploration process; and/or 2) limit the problem to a priori known and/or deterministic transition dynamics with strong smoothness assumptions. Addressing this gap, we propose Analogous Safe-state Exploration (ASE), an algorithm for provably safe exploration in Markov Decision Processes (MDPs) with unknown, stochastic dynamics. Our method exploits analogies between state-action pairs to safely learn a near-optimal policy in a PAC-MDP (Probably Approximately Correct-MDP) sense. Additionally, ASE also guides exploration towards the most task-relevant states, which empirically results in significant improvements in terms of sample efficiency, when compared to existing methods. Source code for the experiments is available at `https://github.com/locuslab/ase`.

1 Introduction

Imagine you are Phillipe Petit in 1974, about to make a tight-rope walk between two thousand-foot-tall buildings. There is no room for error. You would want to be certain that you could successfully walk across without falling. And to do so, you would naturally want to practice walking a tightrope on a similar length of wire, but only a few feet off the ground, where there is no real danger.

This example illustrates one of the key challenges in applying reinforcement learning (RL) to safety-critical domains, such as autonomous driving or healthcare, where a single mistake could cause significant harm or even death. While RL algorithms have been able to significantly improve over human performance on some tasks in the average case, most of these algorithms do not provide any guarantees of safety either during or after training, making them too risky to be used in real-world, safety-critical domains.

Taking inspiration from the tight-rope example, we propose a new approach to safe exploration in reinforcement learning. Our approach, Analogous Safe-state Exploration (ASE), seeks to explore state-action pairs that are analogous to those along the path to the goal, but are guaranteed to be safe. Our work fits broadly into the context of a great deal of recent work in safe reinforcement learning, but compared with past work, our approach is novel in that 1) it guarantees safety during exploration in a stochastic, unknown environment (with high probability), 2) it finds a near-optimal policy in a PAC-MDP (Probably Approximately Correct-MDP) sense, and 3) it guides exploration to focus only on state-action pairs that provide necessary information for learning the optimal policy. Specifically, in our setting we assume our agent has access to a set of initial state-action pairs that are guaranteed to be safe and a function that indicates the similarity between state-action pairs. Our agent constructs an optimistic policy, following this policy only when it can establish that this policy won't lead to a dangerous state-action pair. Otherwise, the agent explores state-action pairs that inform the safety of the optimistic path.

In conjunction with proposing this new approach, we make two main contributions. First, we prove that ASE guarantees PAC-MDP optimality, and also safety of the *entire* training trajectory, with high probability. To the best of our knowledge, this is the first algorithm with this two-fold guarantee in stochastic environments. Second, we evaluate ASE on two illustrative MDPs, and show empirically that our proposed approach sub-

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

stantially improves upon existing PAC-MDP methods, either in safety or sample efficiency, as well as existing methods modified to guarantee safety.

2 Related Work

Safe reinforcement learning. Many safe RL techniques either require sufficient prior knowledge to guarantee safety a priori or promise safety only during deployment and not during training/exploration. Risk-aware control methods (Fleming and McEneaney, 1995; Blackmore et al., 2010; Ono et al., 2015), for example, can compute safe control policies even in situations where the state is not known exactly, but require the dynamics to be known a priori. Similar works (Perkins and Barto, 2002; Hans et al., 2008) allow for learning unknown dynamics, but assume sufficient prior knowledge to determine safety information before exploring. Constrained-MDPs (C-MDPs) (Altman, 1999; Achiam et al., 2017; Taleghan and Dietterich, 2018) and Robust RL (Wiesemann et al., 2013; Lim et al., 2013; Nilim and El Ghaoui, 2005; Ostafew et al., 2016; Aswani et al., 2013), on the other hand, are able to learn the dynamics and safety information, but promise safety only during deployment. Additionally, there are complications with using C-MDPs, such as optimal policies being stochastic and the constraints only holding for a subset of states (Taleghan and Dietterich, 2018).

Other works (Wachi et al., 2018; Berkenkamp et al., 2017; Turchetta et al., 2016; Akametalu et al., 2014; Moldovan and Abbeel, 2012) do consider the problem of learning on unknown environments while also ensuring safety throughout training. Indeed, both our work and these works rely on a notion of similarity between state-action pairs in order to gain critical safety knowledge. For example, (Wachi et al., 2018; Berkenkamp et al., 2017; Turchetta et al., 2016; Akametalu et al., 2014) make assumptions about the regularity of the transition or safety functions of the environment, which allows them to model the uncertainty in these functions using Gaussian Processes (GPs). Then, by examining the worst-case estimate of this model, they guarantee safety on continuous environments. We also refer the reader to a rich line of work outside of the safety literature that has studied similarity metrics in RL, in order to improve computation time of planning and sample complexity of exploration (Givan et al., 2003; Taylor et al., 2009; Abel et al., 2017; Kakade et al., 2003; Taïga et al., 2018) (see Appendix B.6 for more discussion).

However, there are two key differences between the above works (Wachi et al., 2018; Berkenkamp et al., 2017; Turchetta et al., 2016; Akametalu et al., 2014; Moldovan and Abbeel, 2012) and ours. First, (with the exception of Wachi et al. (2018)), these approaches are not reward-directed, and instead focus on only exploring the state-action space as much as possible. Second, although the GP-based methods (Wachi et al., 2018; Berkenkamp et al., 2017; Turchetta et al., 2016; Akametalu et al., 2014) help capture uncertainty, they do not model inherent stochasticity in the environment and instead assume the true transition function to be deterministic. Accounting for this stochasticity presents significant algorithmic challenges (see Appendix B). To the best of our knowledge, Moldovan and Abbeel (2012) is the only work that tackles learning on environments with unknown, stochastic dynamics. Their method guarantees safety by ensuring there always exists a policy to return to the start state. They do, however, assume that the agent knows a-priori a function that can compute the transition dynamics given observable attributes of the states. Our method, however, does not assume known transition functions. Instead it learns the dynamics of state-actions that it has established to be safe, and extends this knowledge to potentially unsafe state-actions.

PAC-MDP learning. Sample efficiency bounds for RL fall into two main categories: 1) regret (Jaksch et al., 2010) and 2) PAC (Probably Approximately Correct) bounds (Strehl et al., 2009; Fiechter, 1994). For our analysis we use the PAC, specifically PAC-MDP (Strehl et al., 2009), framework. PAC-MDP bounds bound the number of ϵ-suboptimal steps taken by the learning agent. PAC-MDP bounds have been shown for many popular exploration techniques, including R-Max (Brafman and Tennenholtz, 2002) and a slightly modified Q-Learning (Strehl et al., 2006). While R-Max is PAC-MDP, it explores the state-action space exhaustively, which can be inefficient in large domains. Another PAC-MDP algorithm, Model-Based Interval Estimation (MBIE) (Strehl and Littman, 2008), outperforms the sample efficiency of R-Max by only exploring states that are potentially along the path to the goal. Our work seeks to extend this algorithm to safety-critical domains and guarantee safety during exploration.

3 Problem Setup

We model the environment as a Markov Decision Process (MDP), a 5-tuple $\langle S, A, R, T, \gamma \rangle$ with discrete, finite sets of states S and actions A, a *known, deterministic* reward function $R : S \times A \to \mathbb{R}$, an *unknown, stochastic* dynamics function $T : S \times A \to \mathcal{P}_S$ which maps a state-action pair to a probability distribution over next states, and discount factor $\gamma \in (0, 1)$. We assume the environment has a fixed initial state and denote it by s_{init}. We also assume that the rewards are known a-priori and bounded between -1 and 1; the rewards that are negative denote dangerous state-

actions.[1] This of course means that the agent knows a priori what state-actions are "immediately" dangerous; but we must emphasize that the agent is still faced with the non-trivial challenge of learning about other a priori unknown state-actions that can be dangerous in the long-term – we elaborate on this in the "Safety" section below. Also note that while most RL literature does not assume the reward function is known, we think this is a reasonable assumption for many real-world problems where reward functions are constructed by engineers. Moreover, this assumption is not uncommon in RL theory (Szita and Szepesvári, 2010; Lattimore and Hutter, 2012).

Analogies. As highlighted in Turchetta et al. (2016), some prior knowledge about the environment is required for ensuring the agent never reaches a catastrophic state. In prior work, this knowledge is often provided as some notion of similarity between state-action pairs; for example, kernel functions used in previous work employing GPs to model dynamics or Lipschitz continuity assumptions placed on the dynamics. Intuitively, such notions of similarity can be exploited to learn about unknown (and potentially dangerous) state-action pairs, by exploring a "proxy" state-action pair that is sufficiently similar and known to be safe. Below, we define a notion of similarity, but the key difference between this and previous formulations is that ours also applies to stochastic environments. Specifically, we introduce the notation of *analogies* between state-action pairs. More concretely, the agent is given *an analogous state function* $\alpha : (S \times A \times S) \times (S \times A) \to S$ and a *pairwise state-action distance mapping* $\Delta : (S \times A) \times (S \times A) \to [0, 2]$ such that, for any $(s, a, \tilde{s}, \tilde{a}) \in (S \times A) \times (S \times A)$

$$\sum_{s' \in S} |T(s, a, s') - T(\tilde{s}, \tilde{a}, \alpha(\cdots, s'))| \leq \Delta((s, a), (\tilde{s}, \tilde{a}))$$

where $\alpha(\cdots, s') = \alpha(s, a, s', \tilde{s}, \tilde{a})$ represents, intuitively, the next state that is "equivalent" to s' for $(\tilde{s}, \tilde{a})$. [2] In other words, for any two state-action pairs, we are given a bound on the L_1 distance between their dynamics: one that is based on a mapping between analogous next states. The hope is that α can provide a much more useful analogy than a naive identity mapping between the respective next states.

To provide some intuition, recall the tight-rope walker example mentioned in the introduction: The tight-rope walker agent must cross a dangerous tight-rope, but wants to guarantee it can do so safely. In this situation, the agent has a "practice" tight-rope (that's only a few feet off the ground) and a "real" tight-rope. In this example, if the agent is in a particular position and takes a particular action on either tight-rope, the change in its position will be the same regardless of what rope it was on.[3] This analogy between the two ropes can be mathematically captured as follows. Consider representing the agent's state as a tuple of the form (prac, x) or (real, x) where the first element denotes which of the two ropes the agent is on, and the second element denotes the position within that rope. Then for any action a, we can say that $\alpha((\text{prac}, x), a, (\text{prac}, x'), (\text{real}, x), a) = (\text{real}, x')$ and $\Delta(((\text{prac}, x), a), ((\text{real}, x), a) = 0$. This would thus imply that by learning a model of the dynamics in the practice setting, and the corresponding optimal policy, an agent would still be able to learn a policy that guaranteed safety even in the real setting.

State-action sets. For simplicity, for any set of state-action pairs $Z \subset S \times A$, we say that $s \in Z$ if there exists any $a \in A$ such that $(s, a) \in Z$. Also, we say that (s, a) is an **edge** of Z if $(s, a) \notin Z$ but $s \in Z$. We use $\mathrm{P}[\cdot|\pi]$ to denote the probability of an event occurring while following a policy π.

Definition 1. *We say that $Z \subset S \times A$ is **closed** if for every $(s, a) \in Z$ and for every next s' for which $T(s, a, s') > 0$, there exists a' such that $(s', a') \in Z$.*

Intuitively, if a set Z is closed, then we know that if the agent starts at a state in Z and follows a policy π such that for all $s \in Z$, $(s, \pi(s)) \in Z$, then we can guarantee that the agent never exits Z (see Fact 1 in Appendix A). We will use $\pi \in \Pi(Z)$ to denote that, for all $s \in Z$, $(s, \pi(s)) \in Z$.

Definition 2. *A subset of state-action pairs, $Z \subset S \times A$ is said to be **communicating** if Z is closed and for any $s' \in Z$, there exists a policy $\pi_s \in \Pi(Z)$ such that $\forall s,\ \mathrm{P}[\exists t,\ s_t = s' \ |\pi_{s'}, s_0 = s] = 1$.*

In other words, every two states in Z must be reachable through a policy that never exits the subset Z. Note that this definition is equivalent to the standard definition of communicating when Z is the set of all state-action pairs in the MDP (see Appendix A).

Safe-PAC-MDP. One of the main objectives of this work is to design an agent that, with high probability (over all possible trajectories the agent takes), learns an optimal policy (in the PAC-MDP sense) while also never taking dangerous actions (i.e. actions with negative rewards) at any point along its arbitrarily long,

[1] Note that our work can be easily extended to have a separate safety and reward function, but we have combined them in this work for convenience.

[2] Note that although α is defined for every $(s, a, s', \tilde{s}, \tilde{a})$ tuple, not all such pairs of state-actions need be analogous to each other. In such cases, we can imagine that $\alpha(s, a, s', \tilde{s}, \tilde{a})$ maps to a dummy state and $\Delta((s, a), (\tilde{s}, \tilde{a})) = 1$.

[3] For this example, we assume the dynamics of the agent on the practice and real tight-ropes are identical, but small differences could be captured by making the $\Delta(\cdot, \cdot, \cdot, \cdot)$ function non-zero.

trajectory. This is a very strong notion of safety, but critical for assuring safety for long trajectories. Such a strong notion is necessary in many real-world applications such as health and self-driving cars where a dangerous action spells complete catastrophe.

We formally state this notion of Safe-PAC-MDP below. The main difference between this definition and that of standard PAC-MDP is (a) the safety requirement on all timesteps and (b) instead of competing against an optimal policy (which could potentially be unsafe), the agent now competes with a "safe-optimal policy" (that we will define later). To state this formally, as in Strehl et al. (2006), let the trajectory of the agent until time t be denoted by p_t and let the value of the algorithm $\mathcal{A}$ be denoted by $V^{\mathcal{A}}(p_t)$ – this equals the cumulative sum of rewards in expectation over all future trajectories (see Def 7 in Appendix A).

Definition 3. *We say that an algorithm $\mathcal{A}$ is **Safe-PAC-MDP** if, for any $\epsilon, \delta \in (0, 1]$, with probability at least $1 - \delta$, $R(s_t, a_t) \geq 0$ for all timesteps t and additionally, the sample complexity of exploration i.e., the number of timesteps t for which $V^{\mathcal{A}}(p_t) > V^{\pi^*_{safe}}(p_t) - \epsilon$, is bounded by a polynomial in the relevant quantities, $(|S|, |A|, 1/\epsilon, 1/\delta, 1/(1-\gamma), 1/\tau, H_{com})$. Here, π^*_{safe} is the safe-optimal policy defined in Def. 6, τ is the minimum non-zero transition probability (see Assumption 4) and H_{com} is "communication time" (see Assumption 3).*

We must emphasize that this notion of safety must *not* be confused with the weaker notion where one simply guarantees safety with high probability *at every step* of the learning process. In such a case, for sufficiently long training trajectories, the agent is guaranteed to take a dangerous action i.e., with probability 1, the trajectory taken by the agent will lead it to a dangerous action as $t \to \infty$.

Safety. Since our agent is provided the reward mapping, the agent knows a priori which state-action pairs are "immediately" dangerous (namely, those with negative rewards). However, the agent is still faced with the challenge of determining which actions may be dangerous *in the long run*: an action may momentarily yield a non-negative reward, but by taking that action, the agent may be doomed to a next state (or a future state) where all possible actions have negative rewards. For example, at the instant when a tight-rope walker loses balance, they may experience a zero reward, only to eventually fall down and receive a negative reward. In order to avoid such "delayed danger", below we define a natural notion of a safe set: a closed set of non-negative reward state-action pairs; as long as the agent takes actions within such a safe set, it will never find itself in a position where its only option is to take a dangerous action. Our agent will then aim to learn such a safe set; note that accomplishing this is non-trivial despite knowing the rewards, because of the unknown stochastic dynamics.

Definition 4. *We say that $Z \subset S \times A$ is a **safe set** if Z is* closed *and for all $(s, a) \in Z$, $R(s, a) \geq 0$. Informally, we also call every $(s, a) \in Z$ as a safe state-action pair.*

3.1 Assumptions

We will dedicate a fairly large part of our discussion below detailing the assumptions we make. Some of these are strong and we will explain why they are in fact required to guarantee PAC-MDP optimality in conjunction with the strong form of safety that we care about (being safe on all actions taken in an infinitely long trajectory) in an environment with unknown stochastic dynamics.

First, in order to gain any knowledge of the world safely, the agent must be provided some prior knowledge about the safety of the environment. Without any such knowledge (either in the form of a safe set, prior knowledge of the dynamics, etc) it is impossible to make safety guarantees about the first and subsequent steps of learning. We provide this to the agent in the form of an initial safe set of state-action pairs, Z_0, that is also communicating. We note that this kind of assumption is common in safe RL literature (Berkenkamp et al., 2017; Bıyık et al., 2019). Additionally, we chose to make this set communicating so that the agent has the freedom to roam and try out actions inside Z_0 without getting stuck.

Assumption 1 (Initial safe set)**.** *The agent is initially given a safe, communicating set $Z_0 \subset S \times A$ such that $s_{init} \in Z_0$.*

In the PAC-MDP setting, we care about how well the agent's policy compares to an optimal policy over the whole MDP. However, in our setting, that would be an unfair benchmark since such an optimal policy might potentially travel through unsafe state-actions. To this end, we will first suitably characterize a safe set Z_{safe} and then set our benchmark to be the optimal policy confined to Z_{safe}.

We begin by defining Z_{safe} to be the set of state-action pairs from which there exists some (non-negative-reward) return path to Z_0. Indeed, *returnability* is a key aspect in safe reinforcement learning – it has been similarly assumed in previous work (Moldovan and Abbeel, 2012; Turchetta et al., 2016) and is also very similar to the notion of stability used to define safety in other works (Berkenkamp et al., 2017; Akametalu et al., 2014). Defining Z_{safe} in terms of returnability ensures that Z_{safe} does not contain any "safe islands" i.e., are safe regions that the agent can venture into,

but without a safe way to exit. At a high-level, this criterion helps prevent the agent from getting stuck in a safe island and acting sub-optimally forever. The reasoning for why we need this assumption and traditional PAC-MDP algorithm do not is a bit nuanced and we discuss this in detail in Appendix B.5.

Definition 5. *We define Z_{safe} to be the set of state-action pairs (s, a) such that $\exists \pi_{return}$ for which:*

$$\mathrm{P}\left[\begin{smallmatrix}\exists t \geq 0 \; s.t. (s_t, a_t) \in Z_0 \\ \& \; \forall t, R(s_t, a_t) \geq 0\end{smallmatrix} \; | \pi_{return}, (s_0, a_0) = (s, a)\right] = 1$$

Note that it follows that Z_{safe} is a safe set (see Fact 3 in Appendix A). Additionally, we will assume that Z_{safe} is communicating; note that given that all actions in Z_{safe} satisfy returnability to Z_0, this assumption is equivalent to assuming that all actions in Z_{safe} are also *reachable* from Z_0. This is reasonable since we care only about the space of trajectories beginning from the initial state, which lies in Z_0. Having characterized Z_{safe} this way, we then define the safe optimal policy using Z_{safe}.

Assumption 2 (Communicatingness of safe set)**.** *We assume $Z_{safe} \subset S \times A$ is communicating.*

Definition 6. *π^*_{safe} is a **safe-optimal** policy in that*

$$\pi^*_{safe} \in \arg \max_{\pi \in \Pi(Z_{safe})} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | \pi, s_0 = s_{init}\right].$$

Besides returnability, another important aspect in the safe PAC-MDP setting turns out to be the time it takes to travel between states. In normal (unsafe) reinforcement learning settings, the agent can gather information from any state-action by experiencing it directly. In this setting, on the other hand, not all state-actions can be experienced safely so the agent must indirectly gather information on a state-action pair of interest by experiencing an analogous state-action. Thus, the agent must be able to visit the informative state-action and return to that state-action pair of interest in polynomial time.

To formalize this, we will assume that within any communicating subset of state-action pairs, we can ensure polynomial-time reachability between states, with non-negligible probability. While this assumption is not made in the normal (unsafe) PAC-MDP setting, we emphasize that this assumption applies to a wide variety of real-world problems and is only violated in contrived examples, such as in a random-walk setting. Specifically, to violate this assumption, there must be two state-action pairs in the safe set where the expected number of steps to move from one to the next is exponential in the state-action size. This can only happen in random-walk-like scenarios where moving backwards has an equal (or higher) probability than moving forward, which occur very rarely in the real-world. To make this last statement concrete, imagine a 1D grid where any action the agent takes leads it to either of the adjacent states with equal probability of 1/2. Then, to reach a state that is n steps away with probability 1/2, it would take the agent, in expectation, an exponential number of steps in n, thus not satisfying Assumption 3. However, if the probabilities of moving forward was 3/4 for one action and moving backward was 1/4 for the other action, then the agent could reach a state that is n steps away with probability 1/2 in less than $2n$ steps, satisfying Assumption 3.

Assumption 3. *(Poly-time communicating) There exists $H_{com} = poly(|S|, |A|)$ such that, for any communicating set $Z \subset S \times A$, and $\forall s' \in Z$, there exists a policy $\pi_{s'} \in \Pi(Z)$ for which*

$$\forall s, \; \mathrm{P}_M[\exists t \leq H_{com}, s_t = s' | \pi_{s'}, s_0 = s] \geq 1/2.$$

Our next assumption is about the transition dynamics: we assume that we know a constant such that any transition either has zero probability or is larger than that known constant. This assumption is *necessary* to perform learning under our strict safety constraints in *unknown, stochastic* dynamics. Specifically, given this assumption, we can use finitely many samples to determine the support of a particular state-action pair's next state distribution. This is critical since the agent cannot take an action unless it knows every possible next state that it could land in. Note that, in a finite MDP, such a τ always exists, we simply assume we have a lower-bound on it.

Assumption 4 (Minimum transition probability)**.** *There exists a* known *$\tau > 0$ such that $\forall s, s' \in S$ and $a \in A$, $T(s, a, s') \in [0] \cup [\tau, 1]$.*

Finally, we make an assumption that will help the agent expand its current estimate of the safe-set along its edges. Specifically, note that to establish safety of an edge state-action pair, it is necessary to establish a return path from it to the current safe-set (see Appendix B.5 for a more in-depth reasoning behind this). Motivated by this we assume the following. Consider any safe subset of state-actions Z and a state-action (s, a) at the edge of Z that also belongs to Z_{safe}. Then, we assume that for every state-action pair that is on the path that starts from this edge and returns to Z, there exists an element in Z that is sufficiently similar to that pair. In other words, for *any* safe subset Z of Z_{safe}, this assumption provides us hope that Z can be expanded by exploring suitable state-actions inside Z.

The fact that this allows expansion of any safe subset Z might seem like a stringent assumption. But we must emphasize this is required to show PAC-MDP optimality: to learn a policy is near-optimal with respect

to π^*_{safe}, intuitively, our algorithm must *necessarily* establish the safety of the set of state-actions that π^*_{safe} could visit. Now, depending on what the (unknown) π^*_{safe} looks like, this set could be as large as (the unknown set) Z_{safe} itself. To take this into account, our assumption essentially allows the agent to use analogies to expand its safe set to a set as large as Z_{safe} if the need arises. Without this assumption, the agent might at some point not be able to expand the safe set and end up acting sub-optimally forever. We note earlier works (Turchetta et al., 2016; Berkenkamp et al., 2017; Akametalu et al., 2014) do not make this assumption because they crucially didn't need to establish PAC-MDP optimality.

Assumption 5 (Similarity of return paths)**.** *For any safe set Z such that $Z_0 \subset Z$ and for any $(\tilde{s}, \tilde{a}) \in Z_{safe}$ such that $\tilde{s} \in Z$ and $(\tilde{s}, \tilde{a}) \notin Z$, we know from Assumption 2 that the agent can return to Z_0 (and by extension, to Z) from $(\tilde{s}, \tilde{a})$ through at least one $\pi_{return} \in \Pi(Z_{safe})$. Let $\tilde{Z}$ denote the set of state-action pairs (s, a) visited by π_{return} before reaching Z, in that,*

$$\mathrm{P}_M\left[\exists t \geq 0 \; \substack{(s_t,a_t)=(s,a) \\ \forall t'<t \;\; s_{t'} \notin Z} \;\middle|\; (s_0, a_0) = (\tilde{s}, \tilde{a}), \pi_{return}\right] > 0.$$

We assume that for all $(s, a) \in \tilde{Z} \setminus Z$, there exists $(s', a') \in Z$ such that $\Delta((s, a), (s', a')) \leq \tau/4$.

To provide some intuition behind this assumption, consider again the tight-rope walker example. For this assumption to be satisfied, the agent must be able to safely learn how to return from any point along the real tight-rope. Since the agent has access to a similar practice tight-rope, the agent can learn to safely cross the practice tight-rope, turn around, and return safely. Once the agent has learned this policy, it has established a safe return path from any point along the real tight-rope. Thus, this assumption is satisfied.

4 Analogous Safe-state Exploration

Given these assumptions, we now detail the main algorithmic contribution of the paper, the Analogous Safe-state Exploration (ASE) algorithm, which we later prove is Safe-PAC-MDP. In addition to safety and optimality, we also do not want to exhaustively explore the state-action space, like R-Max (Brafman and Tennenholtz, 2002), as that can be prohibitively expensive in large domains. We want to guide our exploration, like MBIE (Strehl and Littman, 2008), to explore only the state-action pairs that are needed to find the safe-optimal policy. MBIE does this by maintaining confidence intervals of the dynamics of the MDP, and then by following an "optimistic policy" computed using the most optimistic model of the MDP that falls within the computed confidence intervals. We build on this standard MBIE approach and equip it with a significant amount of machinery to meet our three objectives simultaneously: safety, guided exploration, and optimality in the PAC-MDP sense.

Algorithm 1 Analogous Safe-state Exploration $(\alpha, \Delta, m, \delta_T, R, \gamma, \gamma_{\text{explore}}, \gamma_{\text{switch}}, \tau)$

Initialize: $\hat{Z}_{\text{safe}} \leftarrow Z_0$; $n(s,a), n(s,a,s') \leftarrow 0$;
$\overline{Z}_{\text{goal}} \leftarrow S \times A$; $s_0 \leftarrow s_{\text{init}}$.
Compute confidence intervals using Alg 6 (Appendix B) with parameter δ_T and analogy function α, Δ.
Compute $\overline{\pi}_{\text{goal}}, \overline{Z}_{\text{goal}}$, Z_{explore} using Alg 3 and 4 with parameters γ, τ and reward function R.
Compute $\overline{\pi}_{\text{explore}}, \overline{\pi}_{\text{switch}}$ using value iteration (Appendix B.4) with parameters $\gamma_{\text{explore}}, \gamma_{\text{switch}}$.
for $t = 1, 2, 3, \ldots$ **do**

$$a_t \leftarrow \begin{cases} \overline{\pi}_{\text{goal}}(s_t) & \textbf{if } s_t \in \overline{Z}_{\text{goal}} \;\&\; \overline{Z}_{\text{goal}} \subset \hat{Z}_{\text{safe}} \\ \overline{\pi}_{\text{explore}}(s_t) & \textbf{if } \overline{Z}_{\text{goal}} \not\subset \hat{Z}_{\text{safe}} \\ \overline{\pi}_{\text{switch}}(s_t) & \textbf{otherwise}. \end{cases}$$

Take action a_t and observe next state s_{t+1}.
if $n(s_t, a_t) < m$ **then**
$n(s_t, a_t) \mathrel{+}= 1$, $n(s_t, a_t, s_{t+1}) \mathrel{+}= 1$.
Recompute confidence intervals
Expand $\hat{Z}_{\text{safe}}$ using Alg 2 with parameter τ.
Recompute $\overline{\pi}_{\text{goal}}, \overline{Z}_{\text{goal}}$, Z_{explore}, $\overline{\pi}_{\text{explore}}, \overline{\pi}_{\text{switch}}$.

Policies maintained by ASE. ASE maintains and updates three different policies: (a) an optimistic policy $\overline{\pi}_{\text{goal}}$ that seeks to maximize reward – this is the same as the optimistic policy as in standard MBIE computed on M (except some minor differences), (b) an exploration policy $\overline{\pi}_{\text{explore}}$ that guides the agent towards states in a set called Z_{explore} (described shortly) and finally (c) a "switching" policy $\overline{\pi}_{\text{switch}}$ that can be thought of as a policy that aids the agent in switching from $\overline{\pi}_{\text{explore}}$ to $\overline{\pi}_{\text{goal}}$ (by carrying it from Z_{explore} to $\overline{Z}_{\text{goal}}$ as explained shortly) See Appendix B.4 for details on these policies.

Sets maintained by ASE. ASE also maintains and updates three major subsets of state-action pairs. 1) a safe set $\hat{Z}_{\text{safe}}$ which is initialized to Z_0 and gradually expanded over time (using Alg 2). 2) an "optimistic trajectory" set $\overline{Z}_{\text{goal}}$ (computed in Alg 3), which contains all state-action pairs that we expect the agent would visit if it were to follow $\overline{\pi}_{\text{goal}}$ from s_{init} under *optimistic* transitions. 3) an "exploration set" Z_{explore} (computed using Algs 3 and 4) which contains state-action pairs that, when explored, will provide information critical to expand the safe set. Besides these state-action sets, the algorithm also maintains a set of L_1 confidence intervals as detailed in Alg 6 and Appendix B.2. The key detail here is that the interval for a given state-action pair is not only updated using its samples, but also by exploiting the samples seen at any other well-explored state-action pair that is sufficiently similar to the given pair, according to the given analogies.

How ASE schedules the policies. We discuss how, at any timestep, the agent chooses between one of the above three policies to take an action. First, the agent follows $\overline{\pi}_{\text{goal}}$ whenever it can establish that doing so would be safe. Specifically, whenever (a) $\overline{Z}_{\text{goal}} \subset \hat{Z}_{\text{safe}}$ and (b) the current state s_t belongs to $\overline{Z}_{\text{goal}}$, it is easy to argue that following $\overline{\pi}_{\text{goal}}$ is safe (see proof of safety in Theorem 1). On the other hand, when (a) does not hold, the agent follows $\overline{\pi}_{\text{explore}}$. In doing so, the hope is that, it can explore Z_{explore} well and use analogies to expand $\hat{Z}_{\text{safe}}$ until it is large enough to subsume $\overline{Z}_{\text{goal}}$ (which means (a) would hold then). As a final case, assume (a) holds, but (b) does not i.e., $s_t \notin \overline{Z}_{\text{goal}}$. This could happen if the agent has just explored a state-action pair far away from $\overline{Z}_{\text{goal}}$, which subsequently helped establish (a) i.e., $\overline{Z}_{\text{goal}} \subset \hat{Z}_{\text{safe}}$. Here, we use $\overline{\pi}_{\text{switch}}$ to carry the agent back to $\overline{Z}_{\text{goal}}$. Once carried there, both (a) and (b) hold, so it can switch to following $\overline{\pi}_{\text{goal}}$.

Guided exploration. We would like the agent to explore only relevant states by using the optimistic policy as a guide, like in MBIE. This is automatically the case whenever the agent explores using the optimistic goal policy $\overline{\pi}_{\text{goal}}$. As a key addition to this, we use the optimistic policy to also guide the exploration policy $\overline{\pi}_{\text{explore}}$. As explained below, we do this by only conservatively populating Z_{explore} based on the optimistic trajectory set $\overline{Z}_{\text{goal}}$. Recall that the hope from exploring Z_{explore} is that it can help $\hat{Z}_{\text{safe}}$ expand in a way that it is large enough to satisfy $\overline{Z}_{\text{goal}} \subset \hat{Z}_{\text{safe}}$. Keeping this in mind, the naive way to set Z_{explore} would be to add all of $\hat{Z}_{\text{safe}}$ to it; this will force us to do a brute-force exploration of the safe set, and consequently aggressively expand the safe set in all directions. Instead of doing so, roughly speaking, we compute Z_{explore} in a way that it can help establish the safety of only those actions that (a) are on the edge of $\hat{Z}_{\text{safe}}$ and (b) also belong to $\overline{Z}_{\text{goal}}$. This will help us conservatively expand the safe set in "the direction of the optimistic goal policy". (Note that all this entails non-trivial algorithmic challenges since we operate in an unknown stochastic environment. Due to lack of space we discuss these challenges in Appendix C.2)

5 Theoretical Results

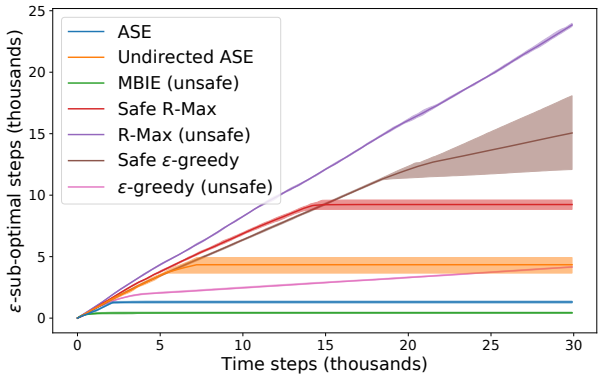
Below we state our main theoretical result, that ASE is Safe-PAC-MDP. We must however emphasize that this result does not intend to provide a tight sample complexity bound for ASE; nor does it intend to compete with existing sample complexity results of other (unsafe) PAC-MDP algorithms. In fact, our sample complexity result does not capture the benefits of our guided exploration techniques – instead, we use practical experiments to demonstrate that these benefits are significant (see Section 6). The goal of this theorem is to establish that ASE is indeed PAC-MDP and safe, which in itself is highly non-trivial as ASE has much more machinery than existing PAC-MDP algorithms like MBIE. In the interest of space we will give a brief overview of the proof & algorithm. A more detailed proof outline can be found in Appendix C. The full (lengthy) proof is included in Appendix D.

Theorem 1. *For any constant* $c \in (0, 1/4]$, $\epsilon, \delta \in (0, 1]$, *MDP* $M = \langle S, A, T, \mathcal{R}, \gamma \rangle$, *for* $\delta_T = \delta/(2|S||A|m)$, $\gamma_{explore} = \gamma_{switch} = c^{1/H}$, *and* $m = O\left((|S|/\tilde{\epsilon}^2) + (1/\tilde{\epsilon}^2)\ln\left(|S||A|/\tilde{\epsilon}\right)\right)$ *where* $\tilde{\epsilon} = \min\left(\tau, \epsilon(1-\gamma)^2, 1/H^2\right)$ *and* $H = O\left(\max\left\{H_{com}\log H_{com}, (1/(1-\gamma))\ln(1/\epsilon(1-\gamma))\right\}\right)$ *ASE is Safe-PAC-MDP with a sample complexity bounded by* $O\left(Hm|S||A|(1/\epsilon(1-\gamma))\ln(1/\delta)\right)$.

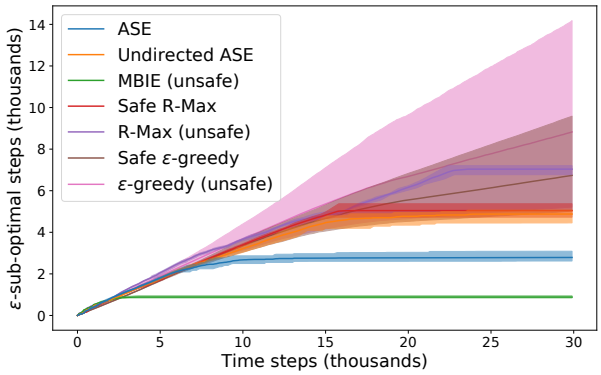
To prove safety, we show in Lemma 3 that Alg 2, which computes $\hat{Z}_{\text{safe}}$, always ensures that $\hat{Z}_{\text{safe}}$ is a safe set. Then, in the main proof of Theorem 1, we argue that the agent always picks state-actions inside $\hat{Z}_{\text{safe}}$. So, it follows that the agent always experiences only positive rewards. Next, in order to prove PAC-MDP-ness, while we build on the core ideas from the proof for PAC-MDP-ness of MBIE (Strehl and Littman, 2008), our proof is a lot more involved. This is because we need to show that all the added machinery in ASE work in a way that (a) the agent never gets "stuck" and (b) whenever the agent takes a series of sub-optimal actions (e.g., while following $\overline{\pi}_{\text{explore}}$ or $\overline{\pi}_{\text{switch}}$), it can "make progress" in some form. As an example of (a), Lemma 4 shows that $\hat{Z}_{\text{safe}}$ is always a communicating set, so the agent can always freely move between the states in $\hat{Z}_{\text{safe}}$. This is critical to show that when the agent follows $\overline{\pi}_{\text{explore}}$ (or $\overline{\pi}_{\text{switch}}$), it can reach Z_{explore} (or $\overline{Z}_{\text{goal}}$) without being stuck anywhere (see Lemma 12). As an example of (b), Lemma 9 and Lemma 10 together show that only informative state-action pairs are added to Z_{explore} i.e., when explored, they *will* help us expand $\hat{Z}_{\text{safe}}$.

6 Experiments

Through experiments, we aim to show that (a) ASE effectively guides exploration, requiring significantly less exploration than exhaustive exploration methods, and (b) the agent indeed never reaches a dangerous state under realistic settings of the parameters (namely m and δ_T in Alg 1). Below, we outline our experiments, deferring the details to Appendix E. For our experiments, we consider two environments. The first is a stochastic grid world containing five islands of grid cells surrounded by "dangerous states" (i.e., states where all actions result in a negative reward). The agent can take actions that allow it to jump over dangerous states to transition between islands and reach the goal state.



(a) Unsafe Grid World

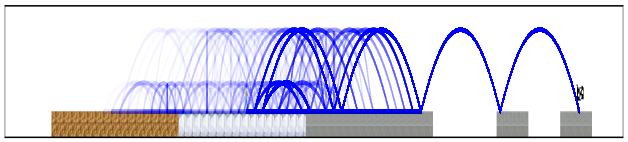


(b) Discrete Platformer

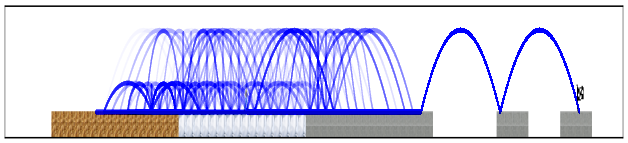
Figure 1: Number of ϵ-sub-optimal steps taken by each agent throughout training. Lines denote averages over five trials and shaded regions mark the max and min.

The second is a stochastic platformer game where certain actions can doom the agent to eventually reaching a dangerous state by jumping off the edge of a platform. In the grid world environment, two state-actions are analogous if the actions are equivalent and the states are near each other (L_∞ distance), and in the stochastic platformer game, if the actions and all attributes but the horizontal position in the state are equivalent and the two states are on the same "surface type".

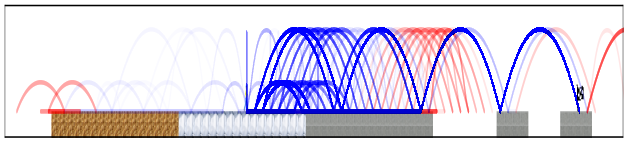
We compare the behavior of our algorithm against both "unsafe" and "safe" approaches to learning reward-based policies. For the unsafe baselines, we consider the original (unsafe) MBIE algorithm (Strehl and Littman, 2008), R-Max (Brafman and Tennenholtz, 2002), and ϵ-greedy, all adapted to use the analogy function (without which, exploring would take prohibitively long). For safe baselines, unfortunately, there is no existing algorithm because no prior work has simultaneously addressed the two objectives of provably safe exploration and learning a reward-based policy in environments



(a) ASE



(b) Safe R-Max



(c) MBIE (unsafe)

Figure 2: All trajectories of different agents on the Discrete Platformer domain. Unsafe trajectories are drawn in red. The brown, white, and grey squares correspond to the different surface types: sand, ice, and concrete, respectively. The agent starts in the center of the leftmost island. The flag represents the goal state.

with unknown stochastic dynamics. To this end, we create safe versions of R-Max and ϵ-greedy (by restricting the allowable set of actions the agent can take to $\hat{Z}_{\text{safe}}$, and using analogies to expand $\hat{Z}_{\text{safe}}$), and also consider an "Undirected ASE," which is a naïver version of ASE that expands $\hat{Z}_{\text{safe}}$ in all directions (not just along the goal policy).

Source code for the experiments is available at `https://github.com/locuslab/ase`.

Results. To measure efficiency of exploration, we count the number of ϵ-sub-optimal steps taken by each agent. To calculate this, we first compute the true safe-optimal Q-function, $Q_M^{\pi^*_{\text{safe}}}$. We then count the number of ϵ-sub-optimal actions taken by the agent, namely the number of times the agent is at a state s_t and takes an action a_t such that $Q_M^{\pi^*_{\text{safe}}}(s_t, a_t) < \max_{a \in A} Q_M^{\pi^*_{\text{safe}}}(s_t, a) - \epsilon$, where $\epsilon = 0.01$. Figure 1 shows our algorithm takes far fewer ϵ-sub-optimal actions before it converges compared to all other *safe* algorithms. As for safety, during our experiments, we observe that, in both domains, the safe algorithms do not reach any unsafe states. In the unsafe grid world domain, the MBIE, R-Max, and ϵ-greedy algorithms encounter an average of 85, 5,016, and 915 unsafe states, respectively, and in the discrete platformer game encounter 83, 542, and 768 unsafe states.

In the platformer domain, as we can see from Figure 2, our method explores only the necessary parts of the initial safe-set, the right side, unlike the Safe R-Max algorithm. Although standard MBIE also directs exploration, it has many trajectories that end in unsafe states, which ASE avoids.

7 Conclusion

We introduced Analogous Safe Exploration (ASE), an algorithm for safe and guided exploration in unknown, stochastic environments using analogies. We proved that, with high probability, our algorithm never reaches an unsafe state and converges to the optimal policy, in a PAC-MDP sense. To the best of our knowledge, this is the first provably safe and optimal learning algorithm for stochastic, unknown environments (specifically, safe during exploration). Finally, we illustrated empirically that ASE explores more efficiently than other non-guided methods. Future directions for the this line of work include extensions to continuous state-action spaces, combining the handling of stochasticity we present here with common strategies in these domains such as kernel-based nonlinear dynamics.

Acknowledgements

Melrose Roderick and Vaishnavh Nagarajan were supported by a grant from the Bosch Center for AI. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE1745016.

References

Wendell H. Fleming and William M. McEneaney. Risk-sensitive control on an infinite time horizon. *SIAM J. Control Optim.*, 1995.

Lars Blackmore, Masahiro Ono, Askar Bektassov, and Brian C. Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE Trans. Robotics*, 2010.

Masahiro Ono, Marco Pavone, Yoshiaki Kuwata, and J. Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. *Auton. Robots*, 2015.

Theodore J. Perkins and Andrew G. Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3, 2002.

Alexander Hans, Daniel Schneegaß, Anton Maximilian Schäfer, and Steffen Udluft. Safe exploration for reinforcement learning. In *ESANN 2008, 16th European Symposium on Artificial Neural Networks, Proceedings*, 2008.

Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, 2017.

Majid Alkaee Taleghan and Thomas G. Dietterich. Efficient exploration for constrained mdps. In *2018 AAAI Spring Symposia*, 2018.

Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

Shiau Hong Lim, Huan Xu, and Shie Mannor. Reinforcement learning in robust markov decision processes. In *Advances in Neural Information Processing Systems*, pages 701–709, 2013.

Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Robust constrained learning-based NMPC enabling reliable mobile robot path tracking. *I. J. Robotics Res.*, 2016.

Anil Aswani, Humberto González, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 2013.

Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. Safe exploration and optimization of constrained mdps using gaussian processes. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, 2018.

Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, pages 908–918, 2017.

Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration in finite markov decision processes with gaussian processes. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, 2016.

Anayo K. Akametalu, Shahab Kaynama, Jaime F. Fisac, Melanie Nicole Zeilinger, Jeremy H. Gillula, and Claire J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control, CDC 2014*, 2014.

Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. In *Proceedings*

of the 29th International Conference on Machine Learning, ICML 2012, 2012.

Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in markov decision processes. *Artificial Intelligence*, 147(1-2):163–223, 2003.

Jonathan Taylor, Doina Precup, and Prakash Panagaden. Bounding performance loss in approximate mdp homomorphisms. In *Advances in Neural Information Processing Systems*, pages 1649–1656, 2009.

David Abel, D Ellis Hershkowitz, and Michael L Littman. Near optimal behavior via approximate state abstraction. *arXiv preprint arXiv:1701.04113*, 2017.

Sham Kakade, Michael J Kearns, and John Langford. Exploration in metric state spaces. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 306–312, 2003.

Adrien Ali Taïga, Aaron Courville, and Marc G Bellemare. Approximate exploration through state abstraction. *arXiv preprint arXiv:1808.09819*, 2018.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

Claude-Nicolas Fiechter. Efficient reinforcement learning. In *Proceedings of the seventh annual conference on Computational learning theory*, pages 88–97. ACM, 1994.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888. ACM, 2006.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

István Szita and Csaba Szepesvári. Constrained policy optimization. In *Model-based reinforcement learning with nearly tight exploration complexity bounds, ICML 2010*, 2010.

Tor Lattimore and Marcus Hutter. Pac bounds for discounted mdps. In *International Conference on Algorithmic Learning Theory*, pages 320–334. Springer, 2012.

Erdem Bıyık, Jonathan Margoliash, Shahrouz Ryan Alimo, and Dorsa Sadigh. Efficient and safe exploration in deterministic markov decision processes with unknown transition models. *arXiv preprint arXiv:1904.01068*, 2019.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.