m-Stage Epsilon-Greedy Exploration for Reinforcement Learning

Rohan Rao, Karthik Narasimhan
Princeton University, Princeton, NJ
Department of Computer Science, Princeton University, Princeton, NJ
{rohanr, karthikn}@princeton.edu

Abstract

Efficient exploration of the environment is a major challenge for reinforcement learning agents, especially in sparse reward settings. This is evident from the fact that simple schemes such as ϵ-greedy remain competitive with more complicated algorithms for exploration. In this paper, we propose a generalization of ϵ-greedy, called m-stage ϵ-greedy in which ϵ increases within each episode but decreases between episodes. This ensures that by the time an agent gets to explore the later states within an episode, ϵ has not decayed too much to do any meaningful exploration. We provide theoretical results motivating the use of our algorithm in task based environments, and provide experimental evidence in two types of environments demonstrating the effectiveness of our method.

Introduction

A central challenge in reinforcement learning is the trade-off between exploration and exploitation. An agent's ability to explore an environment is a key factor in its success, since it must discover actions that provide desirable rewards. At the same time, in order to maximize its expected reward, the agent must exploit its knowledge of 'good' actions to take in different situations. Balancing exploitation with just the right amount of exploration can ensure sufficient returns while allowing for the potential of even higher rewards in the future.

One of the earliest and simplest exploration techniques is ϵ-greedy exploration (Sutton and Barto 2018). The idea is to perform the best possible action according to the learn policy most of the time, but with a probability of ϵ choose an action completely at random. Typically, the ϵ is chosen to be a small value (e.g. 0.05) so as to ensure that the random exploration does not hinder the expected returns of the agent. Recently, deep RL techniques often anneal the value of ϵ across episodes, starting from a high value (e.g. 1) to encourage more exploration at the start of the agent's learning process. Despite its simplicity, ϵ-greedy continues to be competitive with several other more complex schemes that have been proposed in the literature (Taiga et al. 2019; Dabney, Ostrovski, and Barreto 2020).

While the annealing of ϵ ensures that the agent explores sufficiently during the initial episodes, it still falls short in an important aspect. While the agent will explore the initial sub-parts of the Markov Decision Process (MDP), the subsequent reduction of ϵ to a low value will mean that the agent will not perform sufficient exploration in the later stages of the MDP. This would result in a policy that can be learned quickly for the initial parts of an episode, but require exponentially more time to learn for the later parts.

In this paper, we propose m-stage ϵ-greedy, a generalization of ϵ-greedy that allows for dynamically varying the amount of exploration at different points within a single episode. Specifically, we anneal epsilon for the earlier timesteps within an episode before we anneal epsilon for the later timesteps. This allows the agent to *exploit* its (well-learned) policy over the initial parts of the MDP, while still being able to explore the later parts of the MDP where it does not yet have a good policy.

We theoretically analyze our proposed scheme when applied to Q-Learning and show that it is superior to ϵ-greedy, resulting in faster learning. We also perform empirical studies on two different environments – chain MDPs and a LavaWorld environment with multiple rooms. Our experiments support our theoretical expectations and show that our method can result in higher rewards and significantly faster learning. For instance, in the 50-room LavaWorld, our method achieves almost 5x the return compared to vanilla ϵ-greedy.

Related Work

Improvements to ϵ-greedy exploration have been studied through a variety of approaches. One such approach to this involves encouraging exploration through augmenting the game reward with an intrinsic reward that rewards behaviors that explore. Count Based Exploration (Bellemare et al. 2016), (Machado, Bellemare, and Bowling 2018), (Ostrovski et al. 2017), (Tang et al. 2017), Random Network Distillation (Burda et al. 2018), Noisy Networks (Fortunato et al. 2017), and Intrinsic Curiosity (Pathak et al. 2017) are all popular methods to use intrinsic rewards to explore. Each of these approaches aims to reward actions that take the agent to new or surprising states of a game. These approaches have had some success in playing certain games like Montezuma's Revenge with deep RL agents, but in many cases perform worse than ϵ - greedy in practice. (Taiga et al. 2019).

Another approach to the problem of exploration involves the use of "generalized actions" or *options* (Sutton, Precup, and Singh 1999). Of approaches that fall into this category,

(Dabney, Ostrovski, and Barreto 2020) is most relevant to our work. This approach modifies ϵ - greedy exploration to explore by randomly choosing an action and a duration, and repeating the action for that duration. This simple modification to ϵ-greedy exploration helps it escape local optima and do a better job reaching a diverse set of states.

Finally, in tabular environments, there are theoretically motivated exploration schemes such as Thompson Sampling (Thompson 1933) which tries to maximize reward based on randomly drawn beliefs and Upper Confidence Bound (UCB) methods. In general, UCB sampling explores actions which have high potential based on a computed upper confidence bound (Lai and Robbins 1985).

Our algorithm is similar in spirit to UCB as we also try and encourage exploration in regions of the game where our agent is not confident about its policy. Unlike UCB, we implement this by making the assumption that an agent should be more confident in its policy for earlier states than in later states. Additionally, unlike UCB, our exploration scheme is stationary - in that it is independent of the training dynamics of our agent's policy. This property has been hypothesized to lead to successful exploration across a variety of environments (Dabney, Ostrovski, and Barreto 2020). Also, similarly to (Dabney, Ostrovski, and Barreto 2020), our algorithm is a simple to implement generalization of ϵ - greedy exploration.

Background

A Markov Decision Process (MDP) is a 4-Tuple $(\mathcal{S}, \mathcal{A}, \mathbb{T}, \mathcal{R})$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, and for $s \in \mathcal{S}, a \in \mathcal{A}$ $\mathbb{T}(s, a)$ gives a probability distribution over the next state. If $\mathbb{T}$ is not stochastic, we call the MDP deterministic. Finally the reward function $\mathcal{R}(s, a, s')$ assigns a reward for transitioning from s to s' taking action a. Reinforcement Learning algorithms aim to learn a policy for an MDP, $\pi : \mathcal{S} \to \mathcal{A}$ [1] that is optimal in the sense that

$$\pi^* = \arg\max_{\pi} \mathbb{E}\big[\sum_t \gamma^t r_t | \pi\big] \tag{1}$$

Here, γ represents the temporal discount factor applied to our reward.

A popular way to determine π^* is to use Q-learning (Sutton and Barto 1998). Q-learning learns a Q - function, $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ which is defined as

$$Q(s, a) = \max_{\pi} \mathbb{E}[\sum_{i=0}^{\infty} \gamma^i r_{t+i} | s_t = s, a_t = a, \pi] \tag{2}$$

In tabular environments, we can learn a Q function for the states and actions as shown in Algorithm 1. In deterministic environments $\alpha = 1$ is the optimal learning rate.

In this algorithm $\epsilon(t, h)$ determines at each timestep whether the algorithm explores or exploits. A standard choice of $\epsilon(t, h)$ is the following:

[1] Some formulations allow for a stochastic π such that $\pi(s)$ is a probability distribution over $\mathcal{A}$

Algorithm 1: Tabular Q-Learning

$\forall s, a\ Q(s, a) := 0$;
for *episode h* **do**
  reset game;
  s := game.state();
  trajectory = **stack**();
  for *timestep t* **do**
    explore = **flip**($\epsilon(h, t)$);
    if *explore* **then**
      a := game.randomAction();
    else
      a:= $\max_{a' \in \mathcal{A}} Q(s, a')$;
    end
    s', reward = game.play(a);
    trajectory.push((s,a,reward));
    s = s';
  end
  while *trajectory is not empty* **do**
    s, a, reward := trajectory.pop();
    update := reward + $\gamma \max_{a' \in A} Q(s', a')$;
    $Q(s, a) := Q(s, a) + \alpha_{t,h}(\text{update} - Q(s, a))$;
  end
end

Definition 1 (ϵ greedy annealing)**.** *When referring to ϵ greedy exploration, we refer to the following policy. Let H be the number of episodes we anneal during. We define*

$$\epsilon(h, t) = 1 - h\frac{1}{H}.$$

Although our schedule is independent of t, our following results hold even if we anneal within the episode as is common in many implementations.

m-Stage Epsilon Greedy Exploration

We now present a generalization of ϵ-greedy described above, which we refer to as m-Stage ϵ-greedy exploration.

Definition 2 (m-stage ϵ greedy annealing)**.** *When referring to ϵ greedy exploration, we refer to the following policy. Let H be the number of episodes we anneal during. Let T be the steps played per episode. We define*

$$\epsilon^*_m(h, t) = \big[\sum_{j=0}^{m-1} \chi_{\geq j \frac{T}{m}}(t)\big] - h\frac{m}{H}$$

and obtain:

$$\epsilon_m(h, t) = \max(0, \min(1, \epsilon^*(h, t)))$$

where χ_u denotes the indicator variable for the set u.

We note that 1-stage ϵ-greedy exploration is simply ϵ-greedy exploration. Informally, this exploration scheme separates the timesteps of an episode into m segments, and anneals each segment in sequence throughout m stages. Specifically, if we have timesteps $[1, ..., T]$ and choose $m = 2$, segment 1 would be $[1, ..., \frac{T}{2}]$ and segment 2 would be $[\frac{T}{2} + 1, ..., T]$. During stage 1, for the first $\frac{h}{2}$ episodes, ϵ for

the timesteps in segment 1 would get annealed while the epsilon for timesteps in segment 2 would remain unchanged. During stage 2, in the remaining $\frac{h}{2}$ episodes, the epsilon for timesteps in segment 2 would get annealed.

We hypothesize that this annealing schedule will allow an agent to exploit the policy it has for the initial states of an MDP which it has already learned, while still exploring later states.

Experiments

Symmetric and Asymmetric Action Chain

To begin with a few simple starting examples, we consider two MDPs. The first one will be a chain where the agent needs to move right at every step otherwise it will get trapped. The agent gets a reward at every 5 steps it takes to the right up to $5 \cdot k$ steps. This is a known difficult example for vanilla ϵ-greedy exploration. The second one will be an MDP on a line, where the agent begins at 0 and at each timestep chooses to step right or left for $5 \cdot k$ timesteps. The agent gets a reward every time it crosses a multiple of 5 for the first time. Both of these chains are parameterized by k. The second chain is better conditioned for vanilla ϵ-greedy exploration as it has locally symmetric actions (Liu and Brunskill 2019). Informally, an MDP is said to have locally symmetric actions if it is always able to return to its previous state by taking a single action. Although the first MDP does not have locally symmetric actions, our second MDP for the most part does.

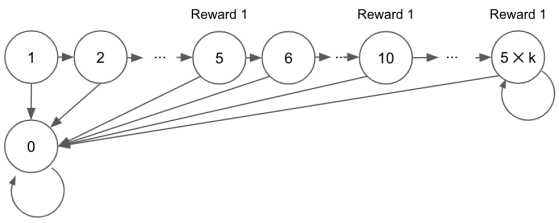

Figure 1: Diagram of a k task asymmetric MDP

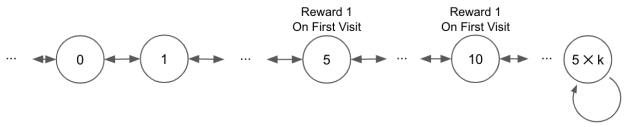

Figure 2: Diagram of a k task symmetric MDP

Although having a locally symmetric action space seems to help ϵ-greedy exploration, in the later states of the environment it still seems like little meaningful exploration is done.

The Floor is Lava

The next environment we consider contains both these elements in the form of a Gridworld composed of a chain of k 5×5 rooms where an agent needs to traverse these

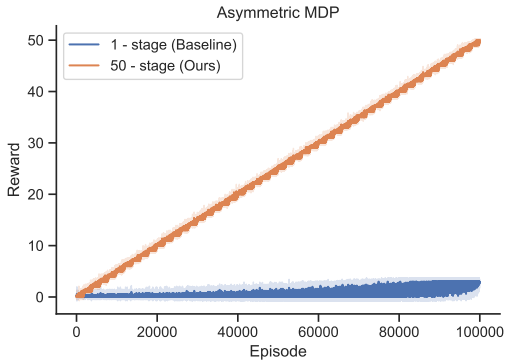

Figure 3: Plot of exploration with an ϵ-greedy policy and with a 50-stage ϵ greedy policy (over 5 runs)

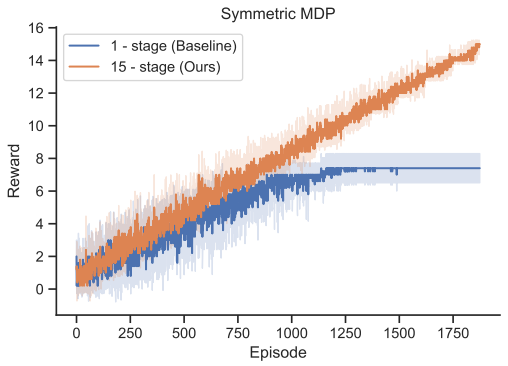

Figure 4: Plot of exploration with an ϵ-greedy policy and with a 15-stage ϵ greedy policy (over 5 runs)

rooms without touching squares of 'lava' within s steps (Figure 5). The action space of this LavaWorld environment is composed of the 8 directions, and with probability 0.01 the agent doesn't move even when taking an action. Each new room the agent sees gives it a reward of 5. This simple setting models many adventure games in which an player needs to explore a world while avoiding hazards that can kill and send the player back to the start. Pitfall and Montezuma's Revenge, two games known for their difficulty in the Atari Learning Environment (Bellemare et al. 2013) include similar challenges.

We plot the reward curves of a Q-Learner equipped with an ϵ-greedy policy and with a m-stage ϵ greedy policy for two such environments, with $k = 5$ in Figure 6 and $k = 50$ in Figure 7. From these experiments, we see that as the number of rooms grow the gap between the performance of our m-stage ϵ greedy agent and our 1-stage ϵ greedy agent grows.

One common feature in these environments is the presence of multiple 'tasks.' In our first two MDPs each task is the set of 5 states before encountering the reward. For the "Floor is Lava" game each room can be considered to be a task. In all of these environments, vanilla ϵ-greedy exploration is unable to complete the later tasks. In the following

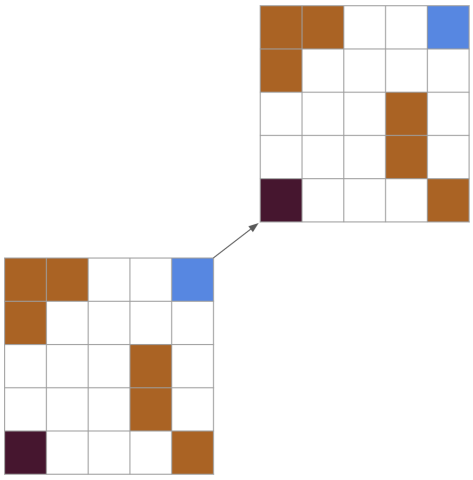

Figure 5: Example rooms of the "Floor is Lava" game. Orange squares represent lava, black represents the room entrance, and blue represents room exit

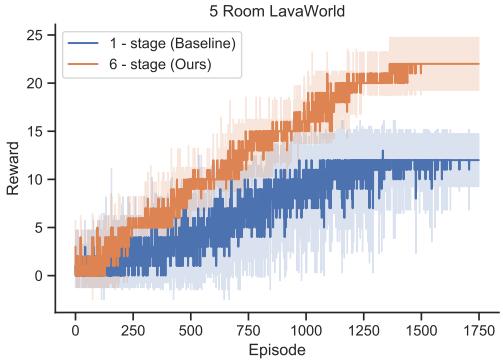

Figure 6: Plot of ϵ-greedy policy and with a 6-stage ϵ greedy policy in a LavaWorld game with $m = 5$ and $s = 30$ (over 5 runs)

section we analyze how our algorithm can succeed in this task-based MDP structure.

Theoretical Analysis

We analyze these two exploration schemes provided in Definitions 1 and 2 to understand situations where ϵ - greedy fails, and how it can be improved. We begin with defining a class of environments with multiple tasks inspired by our experiments.

Definition 3 (Generalized M-Task MDPs)**.** *We consider the class of deterministic* [2] *MDPs $\mathcal{M}$ where each member has an optimal trajectory of length mn $\mathcal{T} = \{a_i\}_{i\in[mn]}$ such that the reward along every prefix of $\mathcal{T}$ which has a length h such that $h \mod n = 0$ has reward greater than any other sequence of actions of length h starting at the initial state of the MDP (task-optimality criterion). We refer to $\{a_{in+1}, ..., a_{(i+1)n}\}$ as the ith generalized task.*

[2]We can construct a similar class of stochastic MDPs but considering the deterministic case simplifies the analysis while preserving the main ideas.

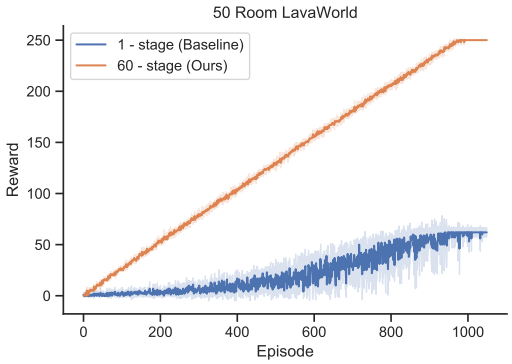

Figure 7: Plot of an ϵ-greedy policy and a 60-stage ϵ greedy policy in a LavaWorld game with $m = 50$ and $s = 300$ (over 5 runs)

Let $\mathcal{A}$ be the cardinality of the action space of $\mathcal{M}$. We say that an algorithm $\mathcal{E}$ solves $\mathcal{M}$ if for every $x \in \mathcal{M}$ using H episodes if $\mathcal{E}$ can determine the optimal trajectory only after experiencing H episodes of the process. We say that $\mathcal{E}$ solves the environment with high probability if the probability (as a function of n and m) that $\mathcal{E}$ does not solve the environment is $o(\frac{1}{m})$. The reason why we choose this high-probability condition is to give us a sense for how many episodes we need to solve many-task environments at arbitrarily high probabilities. Our proof methods in the following section can be adapted to other high-probability type claims.

In this section we prove the following two statements which relate the number of tasks in an environment (m) and the length of each task (n) to the number of episodes (H) needed to train our two algorithms.

Proposition 1. *Epsilon greedy tabular Q-learning requires*

$$H \geq cm^n(\mathcal{A}-1)^n$$

episodes to learn $\mathcal{M}$ with high probability, where $m, n, \mathcal{A}$ are defined in the definition of $\mathcal{M}$ and c is a constant.

Proposition 2. *m - stage epsilon greedy tabular Q-learning requires*

$$H \leq cm(2\mathcal{A})^n \ln(m)$$

episodes to learn $\mathcal{M}$ with high probability, where $m, n, \mathcal{A}$ are defined in the definition of $\mathcal{M}$ and c is a constant.

Under the conditions of Def 3, these propositions motivate why our ϵ schedule leads to faster learning. For example, even assuming a moderately sized n (the length of the task),

the number of episodes required to train ϵ-greedy exploration can be up to a polynomial factor in m higher than for m-stage ϵ-greedy exploration.

Remark 1. *For a Q learner as defined in Algorithm 1, in some environment in $\mathcal{M}$, if the learner experiences some trajectory $\mathcal{T}'$ that is a prefix of the optimal trajectory $\mathcal{T}$ of length h, such that $h \mod n = 0$, it learns the optimal policy for the states in $\mathcal{T}$ along that prefix. This is a consequence of the task-optimality property and the Q-updates in Algorithm 1* [3]*. We will make heavy use of this property as it simplifies analysis and abstracts away the learning dynamics of an agent in stochastic environments.*

More formally let A_i^h be a random variable that represents the action taken by the agent at step i during episode h. Let a_i be ith action in the optimal trajectory. If E_i is the event that we learn the first i tasks, and we train for H episodes then

$$P(E_i) = P(\bigcup_{h\in[H]} \bigcap_{t\in[in]} A_t^h = a_t) \tag{3}$$

Next we proceed to prove the two main propositions.

Proposition 1. *Epsilon greedy tabular Q-learning requires*

$$H \geq cm^n(\mathcal{A}-1)^n$$

episodes to learn $\mathcal{M}$ with high probability, where $m, n, \mathcal{A}$ are defined in the definition of $\mathcal{M}$ and c is a constant.

Proof. For this lower bound, we can construct some $M \in \mathcal{M}$ such that epsilon-greedy requires $H \geq cm^n(\mathcal{A}-1)^n$ episodes to learn M with high probability. We consider a MDP with state space $S = [mn] \cup \{0\}$ and $A = [\mathcal{A}]$ actions. At a given state $s > 0$ if the agent takes action 1, the agent moves to $s+1$, else the agent moves to $s = 0$. At $s = 0$, every action leads back to $s = 0$ (absorbing state). The optimal trajectory then is taking $a = 1$ at every state, and at every nth action on the optimal trajectory the MDP gives a reward of 1, and for all other state action combinations gives reward 0. We illustrate this MDP in Figure 8.

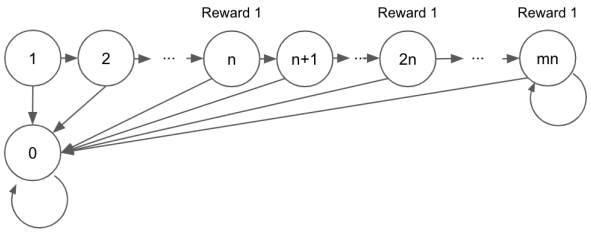

Figure 8: Diagram of MDP in Proposition 1

Let us assume that our agent is initialized knowing the optimal policy for all states $s \leq (m-1)n$ and doesn't know the optimal policy for the remainder of the game. If our agent can learn the game in H episodes that means that the probability that the agent encounters the final reward at state mn is $P(\text{encounter final reward in any episode}) \leq \sum_{h=1}^{H} P(\text{encounter final reward in episode h})$. This union bound gives:

$$\leq \sum_{h=1}^{H} \left[1 - \epsilon(h) + \frac{\epsilon(h)}{\mathcal{A}}\right]^{(m-1)n} \left(\frac{\epsilon(h)}{\mathcal{A}}\right)^n \tag{4}$$

During the first $(m-1)n$ steps with probability $1-\epsilon(h)$ the agent takes the optimal policy, and with probability $\epsilon(h)$ the agent takes a random action and so it takes the right action with probability $\frac{1}{\mathcal{A}}$. The only way to get the final reward is to take the correct action at each step due to the construction of our MDP. Finally for the last n steps the agent has no knowledge of the correct policy and the only way to learn the remainder of the policy is to guess the right string of actions, which just amounts to n independent guesses with success probability $\frac{\epsilon(h)}{\mathcal{A}}$ as there is no intermediate reward between the region of the state space the agent knows, and the final reward. A calculation shows that

$$\max_{\epsilon} \left[1 - \epsilon + \frac{\epsilon}{\mathcal{A}}\right]^{(m-1)n} \left(\frac{\epsilon}{\mathcal{A}}\right)^n = \frac{(m-1)^{(m-1)n}}{m^{mn}(\mathcal{A}-1)^n} \tag{5}$$

$$\leq \frac{1}{m^n(\mathcal{A}-1)^n} \tag{6}$$

at $\epsilon = \frac{\mathcal{A}}{m(\mathcal{A}-1)}$. If we have $H < m^{n-0.9}(m^{0.9}-1)(\mathcal{A}-1)^n$ episodes, then the probability we see the final reward is $< \frac{m^{0.9}-1}{m^{0.9}}$ which means that the probability of failure is $> \frac{1}{m^{0.9}}$ which is not $o(\frac{1}{m})$. This completes the proof. □

Lemma 1. *For an environment some $M \in \mathcal{M}$, while training over H episodes with nm steps per episode, let E_i be the event that an m-stage ϵ greedy agent learns the first i tasks by episode $\frac{i}{m}H$. Then, for $i \in [m]$*

$$P(E_i^c | E_1, ..., E_{i-1}) \leq \prod_{h=1}^{H/m} (1 - \prod_{t=1}^{n} \frac{\epsilon_m(h,t)}{\mathcal{A}}). \tag{7}$$

Proof. Let $a_1, a_2, ..., a_{nm}$ be the optimal trajectory of actions to play the game. let A_j^h be a random variable representing the choice of jth action at episode h. We first note that with our exploration strategy for A_j^h with $h > \frac{i}{m}H$, and $j \leq in$, A_j^h is deterministic. Next let $\widetilde{E}_i$ be the event that our agent has $A_j^h = a_j, j \leq ni$ for some $h \in (\frac{i-1}{m}H, \frac{i}{m}H]$.

$$P(E_i^c | E_1, ..., E_{i-1}) \leq P(\widetilde{E}_i^c | E_1, ..., E_{i-1}) \tag{8}$$

because $\widetilde{E}_i$ implies E_i. We examine the right hand side of this equation. If $h \geq \frac{i-1}{m}$, A_j^h is deterministic for $j \leq (i-1)n$ which means that conditioned on learning the first $(i-1)$ tasks, $A_j^h = a_j$ for $j \leq (i-1)n$ and $h \in (\frac{i-1}{m}H, \frac{i}{m}H]$. This means that for each episode in this range of h, to learn the ith task, our agent just needs to experience $A_j^h = a_j$ for all $j \in ((i-1)n, in]$ in order to experience the whole trajectory $a_1, ..., a_{in}$ and learn the task.

[3] Since this setting is deterministic, the learning rate $\alpha = 1$

Let $h_r = (\frac{i-1}{m}H, \frac{i}{m}H]$, $t_r = ((i-1)n, in]$, then

$$P(\widetilde{E}_i^c | E_1, ..., E_{i-1}) = P((\bigcup_{h \in h_r} \bigcap_{t \in t_r} A_t^h = a_t)^c) \quad (9)$$

$$= P(\bigcap_{h \in h_r} (\bigcap_{t \in t_r} A_t^h = a_t)^c) \quad (10)$$

$$\leq \prod_{h \in h_r} (1 - \prod_{t \in t_r} \frac{\epsilon_m(h,t)}{\mathcal{A}}) \quad (11)$$

Where the last statement comes from the fact that

$$P(A_t^h = a_t | \bigcup_{t', h'} A_{t'}^{h'}) \geq \frac{\epsilon_m(h,t)}{\mathcal{A}} \quad (12)$$

$$t' \neq t, h' \neq h \quad (13)$$

Even when conditioned on any previous learning that has been done. We arrive at our desired statement by observing that

$$\epsilon_m(h,t) = \epsilon_m\big(h - \frac{i-1}{m}H, t - (i-1)n\big) \quad (14)$$

so we can adjust our indices accordingly. □

Proposition 2. *m - stage epsilon greedy tabular Q-learning requires*

$$H \leq cm(2\mathcal{A})^n \ln(m)$$

episodes to learn $\mathcal{M}$ with high probability, where $m, n, \mathcal{A}$ are defined in the definition of $\mathcal{M}$ and c is a constant.

Proof. Let $\{E_i\}_{i \in [m]}$ be the events that the agent learns the first i tasks in some $M \in \mathcal{M}$ over $\frac{i}{m}H$ episodes with nm timesteps per episode. $P(E_m^c)$ is the probability that the agent fails to learn the optimal policy for the environment (E_m is equivalent to the agent learning the environment). The following chain of inequalities holds

$$P(E_m^c) = P((\bigcap_{i \in [m]} E_i)^c) \quad (15)$$

$$\leq \sum_{i \in [m]} P(E_i^c | E_1, ...E_{i-1}) \quad (16)$$

$$\leq m \max_i P(E_i^c | E_1, ..., E_{i-1}) \quad (17)$$

The first statement is true because if any E_i does not occur, then E_m cannot occur. The second statement is a union bound. If we train for H episodes it follows that

$$P(E_i^c | E_1, ..., E_{i-1}) \leq \prod_{h=1}^{H/m} (1 - \prod_{t=1}^{n} \frac{\epsilon_m(h,t)}{\mathcal{A}}) \quad (18)$$

$$\leq \prod_{h=1}^{\frac{H}{2m}} (1 - \frac{1}{(2\mathcal{A})^n}) \quad (19)$$

$$= (1 - \frac{1}{(2\mathcal{A})^n})^{\frac{H}{2m}} \quad (20)$$

$$\leq c_0 e^{-\frac{H}{m(2\mathcal{A})^n}} \quad (21)$$

where the first inequality is just Lemma 1, the second inequality observes that in half of the episodes, $\epsilon(h,t) \geq \frac{1}{2}$. Choosing $H = c_1 m \ln(m)(2\mathcal{A})^n$ for large enough fixed c_1 makes this probability $\leq \frac{1}{m^2}$. And so

$$P(E_m^c) \leq m\frac{1}{m^2} = \frac{1}{m} \quad (22)$$

as desired. □

Propositions 1 and 2 motivate the idea that as the number of stages m of $\mathcal{M}$ grow, for large enough n, we should see an improvement in episode sample complexity between these two algorithms.

In the proof of Proposition 1, we constructed an MDP that forced worst-case behavior for ϵ-greedy exploration, but it is worth considering how realistic having this sort of structure in an MDP is in practice. MDPs that have mechanics where a player can 'die' and then get reset to the beginning of the game generally pose challenging exploration problems to an ϵ-greedy agent (Liu and Brunskill 2019). In the proof of Proposition 1 the MDP we constructed was an extreme case of this - where every misstep sends the agent to an absorbing state until the end of the episode.

A structural property of MDPs that makes them well conditioned for exploration with ϵ-greedy exploration is having *locally symmetric actions* (Liu and Brunskill 2019). Informally, having locally symmetric actions means that an exploration policy can explore the states of the MDP almost like exploring an undirected graph. Many well known easy reinforcement learning environments like simple Gridworlds have this property.

In practice, many task-based environments like Montezuma's revenge, or Text Adventure games have some of both of these elements and so in our experiments we design, many (but not all) states have locally symmetric action spaces but it is possible to die and be forced to restart before all the tasks have been completed.

Although our analysis assumed that we set the number of stages for exploration equal to the number of tasks, our experiments were conducted with more stages than tasks and still showed a substantial improvement over vanilla ϵ-greedy exploration. In our environments, knowing the number of tasks was relatively simple, but for more complicated games it may be harder to determine the number of tasks. In our experiments the improvements we observed did not require much fine-tuning of our choice of m and it would be interesting to see if this also holds true in more complicated environments.

Discussion

Our theoretical and experimental results confirm our hypothesis that ϵ-greedy exploration is poorly conditioned for environments consisting of sequences of tasks. More generally, vanilla ϵ-greedy exploration has trouble exploring the states of an MDP later in the optimal trajectory as ϵ has decayed too much to do meaningful exploration by the time the agent starts encountering those states. Our generalization of ϵ-greedy exploration works to reduce this effect both theoretically and empirically in tabular environments. Finally,

this generalization is easy to implement and reason about and can either serve as an orthogonal improvement to existing agents and can serve as a harder-to-beat baseline in many task-based games.

Future work could extend this exploration scheme to deep reinforcement learning agents. Many continuous environments admit a task-based structure and could benefit from m-stage ϵ greedy exploration. Additionally it would be an interesting problem to examine how this scheme adapts to tasks of different lengths. Finally, a natural extension of our ϵ annealing schedule would involve smoothly increasing epsilon throughout the episode rather than in stages which could be an interesting variation of our exploration scheme.

References

Bellemare, M.; Srinivasan, S.; Ostrovski, G.; Schaul, T.; Saxton, D.; and Munos, R. 2016. Unifying count-based exploration and intrinsic motivation. In *Advances in neural information processing systems*, 1471–1479.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47: 253–279.

Burda, Y.; Edwards, H.; Storkey, A.; and Klimov, O. 2018. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894* .

Dabney, W.; Ostrovski, G.; and Barreto, A. 2020. Temporally-Extended $\epsilon - GreedyExploration$.

Fortunato, M.; Azar, M. G.; Piot, B.; Menick, J.; Osband, I.; Graves, A.; Mnih, V.; Munos, R.; Hassabis, D.; Pietquin, O.; et al. 2017. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295* .

Lai, T. L.; and Robbins, H. 1985. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics* 6(1): 4–22.

Liu, Y.; and Brunskill, E. 2019. When Simple Exploration is Sample Efficient: Identifying Sufficient Conditions for Random Exploration to Yield PAC RL Algorithms.

Machado, M. C.; Bellemare, M. G.; and Bowling, M. 2018. Count-based exploration with the successor representation. *arXiv preprint arXiv:1807.11622* .

Ostrovski, G.; Bellemare, M. G.; Oord, A. v. d.; and Munos, R. 2017. Count-based exploration with neural density models. *arXiv preprint arXiv:1703.01310* .

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 16–17.

Sutton, R.; and Barto, A. 1998. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 9(5): 1054–1054. ISSN 1045-9227. doi:10.1109/tnn.1998.712192. URL http://dx.doi.org/10.1109/TNN.1998.712192.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*.

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112(1): 181–211. ISSN 0004-3702. doi:https://doi.org/10.1016/S0004-3702(99)00052-1. URL http://www.sciencedirect.com/science/article/pii/S0004370299000521.

Taiga, A. A.; Fedus, W.; Machado, M. C.; Courville, A.; and Bellemare, M. G. 2019. On Bonus Based Exploration Methods In The Arcade Learning Environment. In *International Conference on Learning Representations*.

Tang, H.; Houthooft, R.; Foote, D.; Stooke, A.; Chen, O. X.; Duan, Y.; Schulman, J.; DeTurck, F.; and Abbeel, P. 2017. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, 2753–2762.

Thompson, W. R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25(3/4): 285–294.