Policy and Value Transfer in Lifelong Reinforcement Learning

David Abel [†1] **Yuu Jinnai** [†1] **Yue Guo** [1] **George Konidaris** [1] **Michael L. Littman** [1]

Abstract

We consider the problem of how best to use prior experience to bootstrap lifelong learning, where an agent faces a series of task instances drawn from some task distribution. First, we identify the initial policy that optimizes expected performance over the distribution of tasks for increasingly complex classes of policy and task distributions. We empirically demonstrate the relative performance of each policy class' optimal element in a variety of simple task distributions. We then consider value-function initialization methods that preserve PAC guarantees while simultaneously minimizing the learning required in two learning algorithms, yielding MAXQINIT, a practical new method for value-function-based transfer. We show that MAXQINIT performs well in simple lifelong RL experiments.

1. Introduction

The lifelong reinforcement learning (RL) setting formalizes the problem of building agents that must solve a series of related tasks drawn from a task distribution, rather than a single, isolated task. The key question in lifelong RL is the question of *transfer*: how can algorithms exploit knowledge gained by solving previous tasks to improve performance in the next task?

The space of methods for transfer in RL is vast. Prior work has investigated methods for accelerating learning in new environments given partial solutions to related environments, including approaches that incorporate action priors to incentivize guided exploration (Sherstov & Stone, 2005; Rosman & Ramamoorthy, 2012; Abel et al., 2015) and make use of succinct representations that enable efficient inference (Walsh et al., 2006), while others reuse elements of computed policies from related tasks (Fernández & Veloso, 2006; Taylor & Stone, 2007a; Singh, 1992). While a great deal of research has been conducted on understanding effective transfer, the field still lacks a fundamental understanding of what the optimal in-principle approach is in this setting.

We consider the question of how best to initialize an agent's policy or value function for task n, given the optimal policies and value functions obtained by solving tasks 1 through $n-1$. We restrict our attention to two kinds of knowledge: policies and values. We begin with policies, progressing from the simplest setting of constructing the deterministic policy that performs best in expectation for task n, to the stochastic and belief-space policy cases, the latter of which models learning. In the first two cases, we derive the optimal way to initialize the policy for two classes of task distributions. We then turn to value-function initialization, focusing on methods that preserve PAC-MDP guarantees but minimize required learning in R-Max (Brafman & Tennenholtz, 2002) and Delayed Q-Learning (Strehl et al., 2006), both of which preserve PAC bounds via optimistic value-function initialization.

We evaluate each algorithm empirically in a collection of simple lifelong RL tasks. Our empirical and theoretical results show that a practical and simple new method, MAXQINIT, can lower the sample complexity of lifelong learning via value-function-based transfer.

2. Background

Reinforcement learning models an *agent* interacting with an *environment* to maximize long term expected reward (Sutton & Barto, 1998). The environment is typically modeled as a Markov Decision Process (MDP) (Puterman, 2014). An MDP is a five tuple: $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma \rangle$, where $\mathcal{S}$ is a finite set of states; $\mathcal{A}$ is a finite set of actions; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto [0, \text{RMAX}]$ is a reward function, with a lower and upper bound 0 and RMAX; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \Pr(\mathcal{S})$ is a transition function, denoting the probability of arriving in state $s \in \mathcal{S}$ after executing action $a \in \mathcal{A}$ in state $\mathcal{S}$; and $\gamma \in [0, 1)$ is a discount factor, expressing the agent's preference for immediate over delayed rewards.

The agent's action selection strategy is modeled by a *policy*, $\pi : \mathcal{S} \times \mapsto \Pr(\mathcal{A})$, mapping states to rewards. Its goal is to

[†]Equal contribution. [1]Department of Computer Science, Brown University, Providence, RI 02912. Correspondence to: David Abel <david_abel@brown.edu>, Yuu Jinnai <yuu_jinnai@brown.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

Π	$\mathcal{R} \sim D$	$G \sim D$
$\Pi_d : \mathcal{S} \mapsto \mathcal{A}$	Avg. MDP (Ramachandran & Amir, 2007)	(Singh et al., 1994)
$\Pi_s : \mathcal{S} \mapsto \Pr(\mathcal{A})$	Avg. MDP	(Singh et al., 1994)
$\Pi_b : \mathcal{S} \times \Pr(\mathcal{M}) \mapsto \mathcal{A}$	Belief MDP (Åström, 1965)	Belief MDP

Figure 1: A summary of optimality across learning settings and policy classes.

select actions that maximize long term expected reward. A policy is evaluated using Bellman Equation, giving the long term expected reward received by executing that policy:

$$V^\pi(s) = \mathcal{R}(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{T}(s, \pi(s), s') V^\pi(s'). \quad (1)$$

The above measure is known as a *value function*. Also of interest is the *action–value* function, which denotes the value of taking action a and thereafter following policy π:

$$Q^\pi(s, a) = \mathcal{R}(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') V^\pi(s'). \quad (2)$$

We denote $\pi^* = \arg\max_\pi V^\pi$, $V^* = \max_\pi V^\pi$, and $Q^* = \max_\pi Q^\pi$ as the optimal policy, value function, and action-value function respectively. Lastly, we suppose RMAX is a known upper bound on the range of the reward function, and let $\text{VMAX} = \frac{\text{RMAX}}{1-\gamma}$ denote a theoretical upper bound on the maximum possible value achievable in any MDP.

2.1. Lifelong Reinforcement Learning

The goal of our work is to clarify what exactly should be transferred—either as an initial policy, or an initial value function—to maximize performance in lifelong RL. In lifelong RL, an agent solves a series of tasks, rather than a single MDP, and should use its prior experience solving earlier tasks to improve performance in later tasks. We here adopt the lifelong learning framework used by Brunskill & Li (2015; 2014); Isele et al. (2016) and Wilson et al. (2007), inspired by the early work by Thrun & Schwartz (1993):

Definition 1 (Lifelong RL): *Let $\tilde{M} = \mathcal{S}, \mathcal{A}$ be two fixed sets representing a state and action space, respectively. Let D denote a fixed but unknown distribution over $(\mathcal{R}, \mathcal{T}, H, \rho_0)$ quadruples, where $\mathcal{R}$ is a reward function, $\mathcal{T}$ is a transition function, H is a horizon, and ρ_0 is an initial state probability distribution.*

The **lifelong reinforcement learning** *problem consists of the repetition of the following two steps:*

1. *The agent samples a task $(\mathcal{R}, \mathcal{T}, H, \rho_0) \sim D$.*
2. *The agent interacts with the MDP defined by $\tilde{M} \cup (\mathcal{R}, \mathcal{T}, H, \rho_0)$ for H steps.*

The key question in lifelong RL is what knowledge the agent should capture from the tasks it has already solved to import into the task it must solve next. Naturally, lifelong RL is intimately connected to several other problem settings, such as multitask RL and transfer in RL. For further exposition of these settings, see Taylor et al. (2009) and Brunskill & Li (2015). We restrict our attention to subclasses of MDP distributions by making structural assumptions about which MDP constituents may change throughout elements of the support of the task distribution. These subclasses aim to capture distinct types of environments typically studied in the RL literature. Naturally, understanding the setting in its full generality is a direction for future work.

1. $\mathcal{R} \sim D$. In the simplest case, we suppose that *only* the reward function changes over the distribution: $\mathcal{T}$ and ρ_0 do not vary. This variant captures worlds in which tasks or preferences change, but the environment's dynamics remain fixed.

2. $G \sim D$. Perhaps the most common RL tasks are those in which the reward function is *goal based*. That is, every reward in the support of the environmental distribution can be represented as:

$$\mathcal{R}_p(s, a) = \begin{cases} 1 & p(s, a) \\ 0 & \neg p(s, a), \end{cases} \quad (3)$$

for some predicate on state-action pairs, p. Further, states that satisfy goal predicate $p(s, a)$ are terminal: the agent transitions to an absorbing state wherein all actions lead to self-loops forever after.

Note that due to the terminal nature of goals, a change in goal across tasks in the distribution actually changes the transition dynamics across tasks; the assignment of a particular predicate p dictates which states have transitions to an absorbing state. Since the absorbing states change, the $G \sim D$ setting is not a subclass of the $\mathcal{R} \sim D$ setting.

These two settings are simplifications of the full lifelong RL problem, but they offer a general vantage from which to study the nature of optimal transfer in lifelong RL.

Lastly, we define the jumpstart as "The initial performance of an agent in a target task [as] improved by transfer from a source task" (Taylor et al., 2007).

2.2. Related Work

The space of prior work on transfer for RL is broad, encompassing methods for exporting representations, skills or partial policies, and forms of knowledge that bias action selection. We here provide a brief survey of this literature focused on literature dealing with transferring policies, actions priors, and shaping. We do not cover the full details of representation transfer (Walsh et al., 2006; Abel et al., 2017; Taylor & Stone, 2007b) and skill transfer (Konidaris & Barto, 2007; Topin et al., 2015; Pickett & Barto, 2002), but note that these are active and relevant areas of research.

2.2.1. Policy Transfer

Most directly relevant to our agenda are those methods that extract pieces of policies to be used in future tasks. Many of these papers study an agent in a lifelong transfer setting closely aligned with our own. For instance, Fernández & Veloso (2006) developed Policy Reuse, a method for saving chunks of policies for transfer to other related tasks. They inject the effects of prior policies through several strategies, including an ε-greedy-like action selection that uses a prior policy ε of the time. Similarly, Singh (1992) introduces one of the earliest forms of algorithms designed for solving a sequence of related RL tasks. We build on this work by focusing on simple theory that addresses the open question: what should we be transferring in lifelong RL?

2.2.2. Action Priors and Shaping

A key part of the transfer learning literature focuses on methods for biasing an agent's initial action selection strategy, either through a shaping function or action prior.

Sherstov & Stone (2005) introduced one of the earliest instances of action priors in the form of action pruning. They employ action pruning at the task level, focusing on large action-space environments where the reduction in the dimensionality of the action space is dramatic. Along the same lines, Rosman & Ramamoorthy (2012) introduced a method for action-prior transfer in RL that can be incorporated into learning to bias exploration. Abel et al. (2015) propose *goal-based action priors*–computed given access to a training task distribution–which dynamically prune away actions on a complex target task, thereby narrowing the search for a good policy conditioned on a descriptor of the goal and the current state.

Shaping in various forms has long been studied as a mechanism for providing agents with heuristics to accelerate learning, such as the optimal-policy-preserving potential-based shaping (Ng et al., 1999). Mann & Choe (2013) perform transfer between tasks using the assumption that state-action values (in a source and target task) are correlated. They prove that their introduced method of transferring value, based on a task-to-task mapping, cannot hurt the sample complexity of learning, and show that often sample complexity is reduced using the transferred knowledge. The main difference between our work is that they focus on positive transfer to a single test task in contrast to our lifelong setting, and suppose access to a inter-task mapping. Konidaris & Barto (2006) consider an agent facing a sequence of goal-based tasks guaranteed to share some inherent structure, such as containing the same objects or offering the same transition dynamics. In this sense, the problems they consider parallel our $\mathcal{R} \sim D$ and $G \sim D$ settings. They propose learning a shaping function to expedite learning on future tasks. There is a natural parallel to the shaping methods we introduce in Section 3; however, their approach assumes the existence of a separate, agent-centric space in which the shaping function is learned, while we make no such assumption.

Brys et al. (2015) further investigated potential shaping, demonstrating that an arbitrary policy can be transferred by injecting the policy into a reward-shaping function. Our method parallels theirs in the sense that our dynamic shaping method injects the jumpstart policies summarized by Table 1 into initializations. By the main result (Wiewiora, 2003), under restricted conditions, potential shaping and Q-function initialization are equivalent. Other extensions include potential-based shaping for model-based RL (Asmuth et al., 2008), arbitrary translation of reward-function-to-potential (Harutyunyan et al., 2015) and dynamic potential shaping (Devlin & Kudenko, 2012).

3. Theoretical Results

We now aim to partially explain the nature of effective transfer in the two introduced restricted lifelong RL settings. Toward this goal, we first study jumpstart policies for lifelong RL through the following question: which single policy maximizes expected value with respect to the distribution of tasks? That is, if a learner had to start off in a new task drawn from the distribution—and wasn't allowed to change after seeing the task—which policy should it choose? All proofs are deferred to the appendix.

Formally, we'd like to solve the *jumpstart objective*:

$$\arg\max_{\pi \in \Pi} \mathbb{E}_{M \sim D} \left[V_M^{\pi}(s_0) \right], \tag{4}$$

for MDP distribution D and choice of policy class Π. In each of the next two sections we provide two kinds of results for both task distributions:

1. **Jumpstart Policy**: The optimal policy from Π_d and Π_s for the given task distribution.

2. **Value Loss**: A lower bound on the value achieved

by running the jumpstart policy identified by the first result.

3.1. $\mathcal{R} \sim D$

The optimal policy for the jumpstart objective from Π_d is given by Ramachandran & Amir (2007):

Theorem 3.1 (Ramachandran & Amir (2007))**.** *Given a distribution $\Pr(\mathcal{R})$ of reward functions for an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, let an average MDP be an MDP $M_{avg} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}_{avg}, \gamma)$ where $\mathcal{R}_{avg}(s, a) = \mathbb{E}_{\mathcal{R}\sim D}[\mathcal{R}(s, a)]$. The optimal fixed policy for an average MDP maximizes $\mathbb{E}_{M\sim D}\left[V^{\pi}_{M}(s_0)\right]$.*

We refer to the optimal policy for the average MDP as the *average MDP policy*, denoted π^*_{avg}. Since there exists a deterministic optimal policy for any MDP (Ross, 1983), there exists a deterministic policy that is optimal among a set of stochastic policies: $\pi = \arg\max_{\pi\in\Pi_s} \mathbb{E}_{M\sim D}\left[V^{\pi}_{M}(s_0)\right]$. Thus the average MDP policy is the optimal jumpstart policy according to Equation 4 for both Π_d and Π_s.

Our next result establishes a tight lower bound on the value that π^*_{avg} achieves.

Theorem 3.2. *For a distribution of MDPs with $R \sim D$,*

$$\mathbb{E}_{M\in\mathcal{M}}[V_M^{\pi^*_{avg}}(s)] \geq \max_{M\in\mathcal{M}} \Pr(M) V^*_M(s). \qquad (5)$$

Intuitively, running π^*_{avg} is at least as effective as just using the optimal policy of the single MDP with maximal expected value in the distribution. In the Appendix we prove that the above bound is in fact tight. We now discuss the same pair of results for the goal-based-task distribution.

3.2. $G \sim D$

As with $\mathcal{R} \sim D$, we first prove which policy from each class maximizes the jumpstart objective. With a slight modification to notation, we can leverage a classic result from Singh et al. (1994):

Theorem 3.3 (Singh et al. (1994))**.** *Given a distribution of reward and transition functions for an MDP $(S, A, \mathcal{T}, \mathcal{R}, \gamma)$, the policy that maximizes the expected total reward is a policy that maximizes:*

$$\sum_{s\in S} \Pr(s \mid \pi) \sum_{M\in\mathcal{M}} \Pr(M \mid s, \pi) V^{\pi}_{M}(s). \qquad (6)$$

Intuitively, the policy which maximizes the average value function, given updates over the policy's state visitation, maximizes performance in the $G \sim D$ setting. We next provide a lower bound on the value of this policy that parallels Theorem 3.2.

Corollary 3.4. *Given a distribution of reward and transition functions for an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, let M_{gavg} be an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{T}_{avg}, \mathcal{R}_{avg}, \gamma)$ where $\mathcal{R}_{avg}(s, a) = \mathbb{E}_{M\in\mathcal{M}}[\mathcal{R}_M(s, a)]$ and $\mathcal{T}_{avg}(s, a, s') = \mathbb{E}_{M\in\mathcal{M}}[\mathcal{T}_M(s, a, s')]$. An optimal policy π^*_{gavg} for M_{gavg} is a policy with a lower bound in $G \sim D$ setting:*

$$\mathbb{E}_{M\in\mathcal{M}}[V_M^{\pi^*_{gavg}}(s)] \geq \min_{M\in\mathcal{M}} \Pr(M) \max_{M'\in\mathcal{M}} \Pr(M') V^*_{M'}(s). \qquad (7)$$

Note that this bound is actually quite easy to achieve: we can simply solve an arbitrary MDP in the distribution to yield a policy with at least this much value. We suspect that the bound is quite loose and plan to tighten it in future work.

We close this section by noting that in both classes of task distribution, the belief-space policy class, $\Pi_b : \mathcal{S} \times \Pr(M) \mapsto \mathcal{A}$ captures optimal behavior that includes the additional value of in-task learning (Åström, 1965). Of course, solving the belief space problem is computationally intractable, but it does offer an informative upper bound on performance. To this end, we illustrate the performance of the optimal belief-space policy in our experiments.

3.3. Learning

So far, our results have focused on fixed policies that optimize the jumpstart objective. However, the more general question is alluring: Given the choice, what knowledge *should* be reused across MDPs drawn from the same distribution, supposing that an arbitrary learning agent is given this prior knowledge? The results from the previous section suggest that the structure of optimal behavior indeed changes as the constraints placed on the environment or policy class change.

When agents learn, however, reusing the jumpstart policies *exactly* is untenable. Contrary to fixed policies, learning agents must *explore* their environment to ensure the data they collect is informative, while maximizing current performance. The requirement to explore fundamentally changes the measure of a good jumpstart policy. Using the surveyed jumpstart policies exactly will lead to poor performance as they do not necessarily cooperate with exploration strategies. Hence, instead of just evaluating according to the jumpstart objective in Equation 4, we now take into account an agent's propensity to explore.

Specifically, we study classes of value- and reward-function initialization methods for model-free and model-based learning that preserve desirable exploration. Intuitively, these methods parallel potential shaping techniques, and under restricted assumptions are identical (Wiewiora, 2003). We show that a particular type of policy prior, implemented through value-function initialization (of value or reward

estimates, depending on the algorithm used), can lead to a reduction in the sample complexity of learning. More formally, we study which types of initialization functions have two critical properties: (1) preserve the PAC-MDP guarantee of an algorithm, and (2) accelerate learning. At first glance, *none* of the jumpstart policies satisfy property (1), as they all fail to satisfy the optimism property that is critical to many PAC-MDP algorithms.

We first recall the classic result, first introduced by Brafman & Tennenholtz (2002) and later generalized into the PAC-MDP framework (Strehl et al., 2009), based on the sample-complexity of exploration (Kakade, 2003):

Theorem 3.5 (Brafman & Tennenholtz (2002))**.** *The algorithm R-Max is PAC-MDP in parameters (ε, δ). That is, given an $\varepsilon \in [0, \text{VMAX}]$ and a $\delta \in (0, 1.0]$, the algorithm will make at most*

$$\tilde{O}\left(\frac{C \cdot S}{\varepsilon^3(1-\gamma)^3}\text{VMAX}^3\right) \tag{8}$$

mistakes ignoring logarithmic terms, where

$$C = |\{(s,a) \in S \times A : U(s,a) \geq V^*(s) - \varepsilon\}|, \tag{9}$$

and $U(s,a)$ is an upper bound of $Q^*(s,a)$.

That is, R-Max will only make a polynomial number of mistakes with high probability. Critically, the number of mistakes depends on the initialization of U. Likewise, Delayed-Q-Learning (henceforth Delayed-Q) is a model-free algorithm with the same property (Strehl et al., 2006).

Theorem 3.6 (Strehl et al. (2006))**.** *Delayed-Q is PAC-MDP in parameters (ε, δ). That is, given an $\varepsilon \in [0, \text{VMAX}]$ and a $\delta \in (0, 1.0]$, the algorithm will make at most:*

$$\tilde{O}\left(\frac{D \cdot \text{VMAX}(1+\gamma\text{VMAX})^2}{\varepsilon^4(1-\gamma)^4}\right) \tag{10}$$

mistakes ignoring logarithmic terms, where

$$D = \sum_{(s,a) \in S \times A} [U(s,a) - V^*(s)]_+, \tag{11}$$

$[x]$ is defined as $\max(0, x)$, and $U(s,a)$ is an upper bound of $Q^*(s,a)$.

These two algorithms will serve as exemplar PAC-MDP model-based and model-free algorithms respectively. The polynomial bounds indicate that these algorithms explore their environments efficiently. Our original transfer objective now becomes: initialize $U(s,a)$ so as to reduce the size of C or D but still preserve the PAC-MDP property.

Our next result summarizes a new method, MAXQINIT, that balances this tradeoff. In particular, we inject prior knowledge, such as the knowledge captured by a jumpstart policy, as an initial value function $U(s,a)$ for both algorithms. A setting of this function can bias or accelerate learning, prune actions, and enforce partial policies on a state-action basis, while simultaneously lowering sample complexity, insofar as C and D are reduced in size. A stronger result might be obtained by combining these updates with a computationally efficient incremental planner, as in Strehl et al. (2012).

We first consider an optimal but highly unrealistic form of initializing U. Let $Q_{\max}$ be a function on state-action pairs that takes the maximum value of Q^* across MDPs in the distribution:

$$Q_{\max}(s,a) = \max_{M \in \mathcal{M}} Q^*_M(s,a). \tag{12}$$

By definition, $Q_{\max}$ is the tightest upper bound of Q^*_M over MDPs the distribution. Thus, $U = Q_{\max}$ minimizes both C and D, and so would serve as an appropriate value.

$Q_{\max}$ is unattainable as it requires prior knowledge about every MDP in the distribution. We propose a more realistic approach called MAXQINIT, a Q-function transfer algorithm that estimates $Q_{\max}$ empirically:

$$\widehat{Q}_{\max}(s,a) = \max_{M \in \widehat{\mathcal{M}}} Q_M(s,a), \tag{13}$$

where $\widehat{\mathcal{M}}$ is the set of MDPs the agent has sampled so far, and Q_M is a Q-function the agent learned from interacting with each MDP. Algorithm 1 provides pseudocode for the updating optimism procedure.

Algorithm 1 MAXQINIT

INPUT: $s_0, t, H, D, \mathscr{A}, \widehat{p}_{\min}, \widehat{Q}_{\max}$
OUTPUT: $U(s,a)$
if $t > \frac{\ln(\delta)}{\ln(1-\widehat{p}_{\min})}$ **then**
  $\forall_{s,a} : Q_t(s,a) \leftarrow \widehat{Q}_{\max}$
else
  $\forall_{s,a} : Q_t(s,a) \leftarrow \text{VMAX}$
end if
$M \leftarrow \text{sample}(D)$
$Q_{\mathscr{A},M} \leftarrow \mathscr{A}(M, s_0, H)$
$\forall_{s,a} : Q_t(s,a) \leftarrow Q_{\mathscr{A},M}(s,a)$
$\forall_{s,a} : Q_{\max} \leftarrow \max\left\{\widehat{Q}_{\max}(s,a), Q_t(s,a)\right\}$
return $\widehat{Q}_{\max}$

We next offer a lower sample bound on MDPs required for preserving the optimism property with high probability.

Theorem 3.7. *Suppose $\mathscr{A}(M, s_0, H)$ produces ε-accurate Q functions for a subset of the state action space given an MDP M, an initial state s_0, and a horizon H. For a given $\delta \in (0, 1]$. Then, after $t \geq \frac{\ln(\delta)}{\ln(1-p_{min})}$ sampled MDPs, for $p_{min} = \min_{M \in \mathcal{M}} \Pr(M)$, the* MAXQINIT *initialization*

will return $\hat{Q}_{max}$ such that for all state action pairs (s, a):

$$\widehat{Q}_{\max}(s,a) \geq \max_{M} Q^*_M(s,a), \tag{14}$$

with probability $1-\delta$.

Consequently, with high probability, updating $U(s,a)$ using the above strategy preserves the optimism property required by the PAC-MDP algorithms. Further, the size of C and D is reduced, as those state–action pairs (s,a) for which $\widehat{Q}_{max}(s,a)$ are below the mistake bound are removed from the algorithm's sample complexity. Our sample bound requires $\widehat{p}_{\min}$, a parameter indicating the lowest probability MDP in the distribution (or an approximation thereof).

In summary, our results serve two purposes. First, we illustrate which individual policies maximize the jumpstart objective to address the simplest version of the question: how should agents begin their learning for task $n+1$, if they've solved some number of tasks $1, \ldots, n$? We accompany these results with lower bounds on the performance of these policies to illustrate their utility. Second, we use the intuition developed by our jumpstart policies to identify a practical transfer method well suited to lifelong learning.

4. Experiments

Our experiments evaluate both the jumpstart policies and initialization methods. Our code is freely available for reproducibility and extension.[1]

4.1. Jumpstart Policies

First, we evaluate jumpstart policies. The goal of these experiments is to showcase the relative performance of different jumpstarts in different lifelong learning settings. In each environment, we compare four policies:

- π_{avg} (●): The optimal policy in the average MDP.
- π_{prior} (■): The action-prior policy, which stochastically chooses actions according to the probability distribution over action optimality: $\pi_{prior}(a \mid s) = \Pr_{M \sim D}\left(a = \arg\max_a Q^*_M(s,a) \;\middle|\; s\right)$
- π^u (◆): The uniform random policy.
- π^*_b (▲): The optimal belief-space policy—the policy that maximizes value over the Belief MDP (Åström, 1965) defined by the MDP distribution.

For each policy, in each experiment, we sample a reward function, run the policy in the resulting MDP for 100 steps and repeat. All plots show average performance over the task distribution, an empirical estimate of our jumpstart objective. We provide 95% confidence intervals over the joint distribution-sampling and policy-execution process (which leads to relatively large intervals in many cases due to the inherent entropy of the task distribution). We set $\gamma = 0.95$ for all experiments.

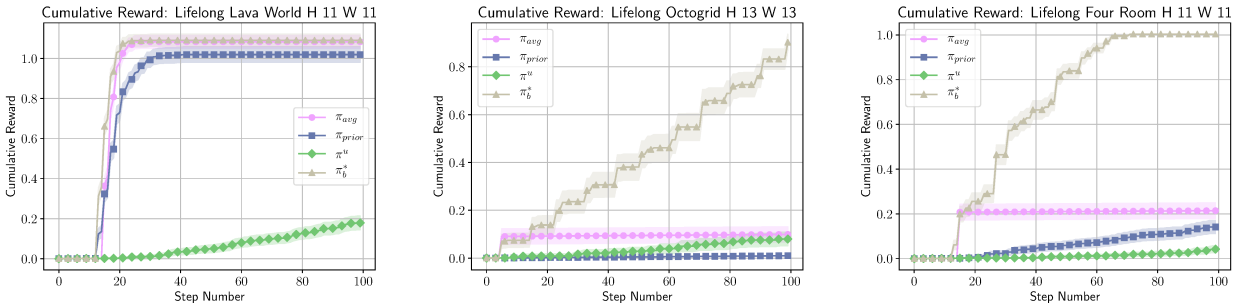
For $\mathcal{R} \sim D$, we use a use-typical 11×11 grid world task distribution with each reward function specifying four to eight lava states placed throughout the grid. Entering a state with lava provides a reward of 0, with all other rewards set to 0.001, apart from states that satisfy the goal predicate, which yields 1.0. The goal location is terminal and fixed across tasks to the top right cell, while the start state is fixed to the bottom left cell. Lastly, there is a 0.1 slip probability, in which the agent moves in one of the two directions (chosen uniformly) perpendicular to its chosen action.

For $G \sim D$, we experiment with two grid-world distributions. In the first, "Octogrid", the agent starts in the center of a 13×13 grid world (see Figure 2 of the Appendix for an image of the problem). The agent is surrounded by twelve hallways, three in each cardinal direction. A single goal appears at the end of one hallway in each G sampled from the distribution, chosen uniformly at random over hallways. All rewards are set to 0 apart from the transition into the goal state, which yields 1.0. The second task is Four Rooms, the classic 11×11 grid-world variant from Sutton et al. (1999). The goal distribution is uniform over each of the furthest corners and the center of the non-starting rooms (for a total of six goals), with the agent starting in the bottom left room. For both of these tasks, there is no slip probability.

Results for the jumpstart experiments are presented in Figure 2. Across all plots, the belief-space policy and the uniform random policy serve as upper and lower bounds on performance. In Lava World, the optimal policy in the average MDP performs better than the uniform random policy, as expected. The average MDP policy performs better than the action-prior policy in Lava World. In both experiments, the average MDP policy has almost no increase after the first 20 steps, but it still performs better than the action-prior policy.

In Octogrid, both the average MDP and the action-prior MDP are worse than the uniform random policy after 60 steps. In the task, the goals are spread out (see Appendix). The average MDP policy is deterministic, so the policy will go to one of the goals, yielding an expected (undiscounted) value of $1/N$, where N is the number of goals/arms. The action prior policy yields even worse behavior: when the agent is down one of the arms on route to the goal, the action prior policy moves away from the goal (toward all other the goals) with probability $(N-1)/N$. Therefore, for a sufficiently long arm, and a high enough N, the action prior

[1] https://github.com/david-abel/transfer_rl_icml_2018.



(a) Lava ($\mathcal{R} \sim D$) (b) Octo ($G \sim D$) (c) Four Room ($G \sim D$)

Figure 2: Results of jumpstart experiments in $\mathcal{R} \sim D$ (top) and $G \sim D$ (bottom) task distributions. Plots show cumulative reward averaged over 500 (Lava and Octo) and 300 (Four Room) MDP samples from D.

policy will reach a goal with exponentially low probability. By contrast, the uniform random policy will reach a goal according to the random walk.

Note the surprising advantage of π^*_{avg} compared to π_{prior}. While relatively negligible over a longer horizon, recall that even a moderate advantage here is indicative of the boost a learning agent receives from its first round of experiences. In domains with sparse and delayed reward, improvement of this kind can lead to dramatic decreases in learning time. Of course, the upper bound provided by the belief policy π^*_b showcases how much room a learning agent has to grow *after* its initial experiences. Even still, early on, π^*_{avg} tracks closely with π^*_b, particularly in the Octogrid.

4.2. Learning

In our second set of experiments, we evaluate methods for value initialization in lifelong RL. We compare Q-learning, R-Max, and Delayed-Q. For Q-Learning, we used ε-greedy action selection with $\varepsilon = 0.1$, and set learning rate $\alpha = 0.1$. For R-Max, we set the knownness threshold to 10, the number of experiences m of a state-action pair before an update to be allowed to 5, the number of value iterations to 5, and for Delayed-Q we set $m = 5$ and a constant exploration bonus $\epsilon_1 = 0.1$. In all experiments, the agent samples a goal uniformly at random, interacts with the resulting MDP for 100 episodes with 100 of steps per episode.

We evaluate each algorithms with four initializations of Q:

1. VMAX: Q is initialized as the maximum of V.
2. $UO(\mathcal{M}, Q^*)$: Q is initialized to be the maximum of optimal Q that could be achieved among all the MDPs.
3. $UO(\widehat{\mathcal{M}}, Q^*)$: Q is initialized to be the maximum of optimal Q that could be achieved among the MDPs in the empirical distribution.
4. MAXQINIT: Q is initialized to be the maximum of estimated Q that could be achieved among the MDPs in the empirical distribution.

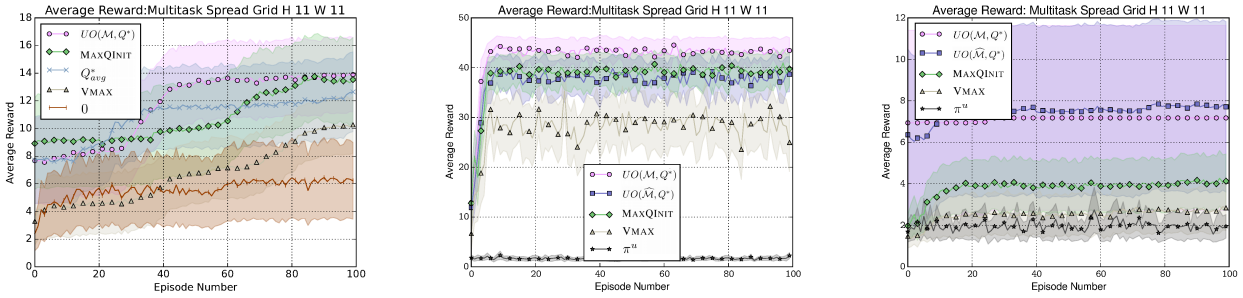
$UO(\mathcal{M}, Q^*)$ and $UO(\widehat{\mathcal{M}}, Q^*)$ are (impractical) idealizations of MAXQINIT where an agent knows the Q^* of every MDP in the true or empirical distribution.

For Q-learning, we also evaluate Q^*_{avg}, where Q is initialized as the average Q^* value over the true distribution (average MDP), and where Q is initialized with 0. We excluded $UO(\widehat{\mathcal{M}}, Q^*)$ for Q-learning as Q-learning agents tend to learn a Q-value close enough to Q^* within a given number of episodes.

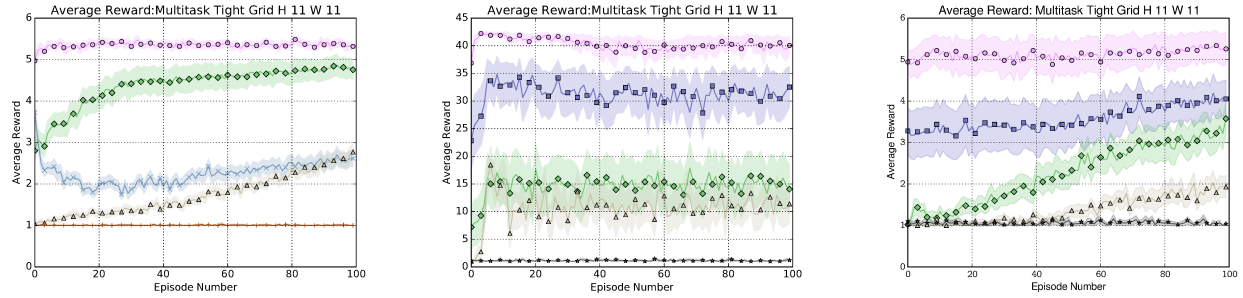
We experiment with each approach on two extremely simple 11×11 gridworld task distributions. In the first ("Tight") the seven possible goals appear in the upper rightmost cells in the square, while in the second ("Spread"), the seven goals can appear near each of the three non-starting corners. Each sampled problem only has a single goal state, chosen uniformly at random among the eight possible cells. All transitions have reward 0 except for transitions into the goal state, which yield $+1$. No slip probability is used. All agents start in the bottom left corner of the grid.

Results for the initialization experiments are presented in Figure 3, with the Tight grid in the top row and Spread grid in the bottom. Notably, Q-learning paired with MAXQINIT outperformed Q^*_{avg} and VMAX. Although Q^*_{avg} is a Q-function of the optimal fixed policy, learning algorithms with optimistic initialization quickly outperformed it. Similarly, for both R-Max and Delayed-Q, MAXQINIT consistently outperformed or matched the performance of VMAX.

Collectively, these empirical findings in suggest that MAXQINIT is a general, principled, and practical means of safely accelerating learning in lifelong RL. Naturally, scaling our approach is a direction of future work.



(a) Q-learning Spread (b) R-Max Spread (c) Delayed-Q Spread



(d) Q-learning Tight (e) R-Max Tight (f) Delayed-Q Tight

Figure 3: Comparison of Q-learning, Delayed-Q, and R-Max with Q-function initialized by VMAX and MaxQInit (and average MDP for Q-learning). Plots show reward averaged over 100 MDP samples from D.

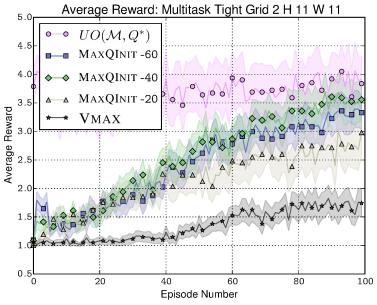


Figure 4: Comparison of Delayed-Q with a varied number of task samples prior evaluation.

4.2.1. Effect of Sample Budget on Performance

In our final experiment, we explore the choice of δ on MaxQInit and the other initialization methods. That is, each algorithm runs $k = \ln(\delta)/\ln(1 - p_{min})$ tasks with the initial Q-function set to Vmax, then we plot the resulting algorithms performance using $\hat{Q}_{\max}$ for 60 tasks. The task is an $11\times$ 11 grid world where the agent starts at the opposite corner. The goal is chosen uniformly among the 21 cells within five moves from the corner. We set $m = 5$, $\epsilon_1 = 0.001$, with the true $p_{min} = \frac{1}{21}$. We test with δ set to each of $\{.35, 0.13, .05\}$, resulting in $k = \{20, 40, 60\}$ task samples to learn from. The results—presented in Figure 4—suggest that even with a few samples, MaxQInit is an effective means of accelerating lifelong RL.

5. Conclusion

This paper addresses the question: which knowledge should be transferred in lifelong RL? We formalize our answer through two simple families of results. First, we derive policies that maximize expected behavior over a distribution of tasks, highlighting subtleties of task distributions that effect decision making. Second, we use these results to introduce MaxQInit, an algorithm for updating initializations in lifelong learning that trades off between tempered optimism and lowering sample complexity. Our experiments explore the performance of the jumpstart policies and showcase the practicality of MaxQInit for accelerating algorithms in lifelong RL in simple domains.

In future work we hope to to generalize these results beyond the limited setting we study here. In particular, we take our findings about policy and value transfer to suggest paths for making optimal use of *options* in lifelong RL, an active focus of recent literature (Brunskill & Li, 2014).

Acknowledgement

We would like to thank Emre Sonmez, whose thoughts on the problem informed the early development of this paper. This research was supported in part by an AFOSR Young Investigator Grant to George Konidaris, under agreement number FA9550-17-1-0124 and supported in part by the DARPA XAI program.

References

Abel, D., Hershkowitz, D., Barth-Maron, G., Brawner, S., O'Farrell, K., MacGlashan, J., and Tellex, S. Goal-based action priors. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling*, pp. 306–314, 2015.

Abel, D., Arumugam, D., Lehnert, L., and Littman, M. Toward good abstractions for lifelong learning. In *Proceedings of the NIPS Workshop on Hierarchical Reinforcement Learning*, 2017.

Asmuth, J., Littman, M., and Zinkov, R. Potential-based shaping in model-based reinforcement learning. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pp. 604–609, 2008.

Åström, K. J. Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

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

Brunskill, E. and Li, L. PAC-inspired option discovery in lifelong reinforcement learning. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, pp. 316–324, 2014.

Brunskill, E. and Li, L. The online discovery problem and its application to lifelong reinforcement learning. *CoRR, abs/1506.03379*, 2015.

Brys, T., Harutyunyan, A., Taylor, M., and Nowé, A. Policy transfer using reward shaping. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 181–188, 2015.

Devlin, S. and Kudenko, D. Dynamic potential-based reward shaping. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pp. 433–440, 2012.

Fernández, F. and Veloso, M. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 720–727, 2006.

Harutyunyan, A., Devlin, S., Vrancx, P., and Nowé, A. Expressing arbitrary reward functions as potential-based advice. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2652–2658, 2015.

Isele, D., Rostami, M., and Eaton, E. Using task features for zero-shot knowledge transfer in lifelong learning. In *25th International Joint Conference on Artificial Intelligence*, pp. 1620–1626, 2016.

Kakade, S. M. *On the sample complexity of reinforcement learning*. PhD thesis, University of London, 2003.

Konidaris, G. and Barto, A. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd International Conference on Machine learning*, pp. 489–496, 2006.

Konidaris, G. and Barto, A. Building portable options: Skill transfer in reinforcement learning. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pp. 895–900, 2007.

Mann, T. and Choe, Y. Directed exploration in reinforcement learning with transferred knowledge. In *European Workshop on Reinforcement Learning*, pp. 59–76, 2013.

Ng, A. Y., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the International Conference on Machine Learning*, volume 99, pp. 278–287, 1999.

Pickett, M. and Barto, A. Policyblocks: An algorithm for creating useful macro-actions in reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 506–513, 2002.

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

Ramachandran, D. and Amir, E. Bayesian inverse reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 51, pp. 1–4, 2007.

Rosman, B. and Ramamoorthy, S. What good are actions? accelerating learning using learned action priors. In *Proceedings of the 2012 IEEE International Conference on Development and Learning and Epigenetic Robotics*, pp. 1–6, 2012.

Ross, S. M. *Introduction to Stochastic Dynamic Programming: Probability and Mathematical*. New York Academic Press, Inc., 1983.

Sherstov, A. and Stone, P. Improving action selection in MDPs via knowledge transfer. In *Proceedings of the*

Twentieth National Conference on Artificial Intelligence, pp. 1024–1029, 2005.

Singh, S., , Jaakkola, T., and Jordan, M. Learning without state-estimation in partially observable markovian decision processes. In *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 284–292, 1994.

Singh, S. P. Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8(3-4): 323–339, 1992.

Strehl, A., Li, L., Wiewiora, E., Langford, J., and Littman, M. PAC model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 881–888, 2006.

Strehl, A. L., Li, L., and Littman, M. L. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

Strehl, A. L., Li, L., and Littman, M. L. Incremental model-based learners with formal learning-time guarantees. In *Proceedings of the conference on Uncertainty in Artificial Intelligence*, 2012.

Sutton, R., , Precup, D., and Singh, S. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1): 181–211, 1999.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Taylor, M. and Stone, P. Cross-domain transfer for reinforcement learning. In *Proceedings of the 24th International Conference on Machine learning*, pp. 879–886, 2007a.

Taylor, M. and Stone, P. Representation transfer for reinforcement learning. In *AAAI 2007 Fall Symposium on Computational Approaches to Representation Change during Learning and Development*, 2007b.

Taylor, M., , and Stone, P. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

Taylor, M. E., Whiteson, S., and Stone, P. Transfer via inter-task mappings in policy search reinforcement learning. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pp. 37. ACM, 2007.

Thrun, S. and Schwartz, A. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*, 1993.

Topin, N., Haltmeyer, N., Squire, S., Winder, J., desJardins, M., and MacGlashan, J. Portable option discovery for automated learning transfer in object-oriented Markov decision processes. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pp. 3856–3864, 2015.

Walsh, T., Li, L., and Littman, M. Transferring state abstractions between MDPs. In *ICML Workshop on Structural Knowledge Transfer for Machine Learning*, 2006.

Wiewiora, E. Potential-based shaping and Q-value initialization are equivalent. *Journal of Artificial Intelligence Research*, 19:205–208, 2003.

Wilson, A., Fern, A., Ray, S., and Tadepalli, P. Multi-task reinforcement learning: a hierarchical Bayesian approach. In *Proceedings of the 24th International Conference on Machine learning*, pp. 1015–1022, 2007.