Exploiting Multiple Levels of Abstractions in Episodic RL via Reward Shaping

Roberto Cipollone, Giuseppe De Giacomo, Marco Favorito, Luca Iocchi, Fabio Patrizi

Sapienza University of Rome

Abstract

One major limitation of Reinforcement Learning (RL) algorithms, which limits applicability in many practical domains, is the large amount of samples required to learn an optimal policy. To improve learning efficiency, we consider a hierarchy of abstraction layers, where the Markov Decision Process (MDP) underlying the target domain can be abstracted at multiple levels by other MDPs. Each abstract model in the hierarchy is a coarser representation of the next one below, which captures the relevant dynamics in finer resolution. This paper proposes a novel form of Reward Shaping defined in terms of the solution obtained in the abstract levels. Theoretical guarantees about optimality and experimental validation of learning efficiency are discussed in the paper. Our technique has minimum requirements in the design of abstract models and is also tolerant to modelling errors in abstractions, thus making the proposed method of practical interest.

Introduction

Reinforcement Learning (RL) agents have no model available to predict outcomes of their actions. While this allowed wide applicability of RL algorithms, this lack of knowledge also demands a significant number of interactions with the environment before an optimal policy can be estimated. Indeed, most of the successes of RL achieved in recent years come from the digital world (e.g. video games, simulated environments), where a large amount of samples can be easily generated. Still, even in these cases, such large number of samples might not be available, as the simulation costs may be very high. As a result, applications of RL in real environments, such as real robots, are still very rare.

Many RL tasks are goal-oriented, in which a set of environment states are denoted as target configurations. Complex tasks induce sparse goals and, as a consequence, sparse rewards. This is known to be a challenging scenario for RL, which increases the requirements on the number of samples to collect. Unfortunately, sparse goal states are very common, as they may arise in simple tasks on large state spaces (such as reaching specific locations in a complex environment), or complex behaviours even in modest environments (such as the successful completion of a desired sequence (Brafman, De Giacomo, and Patrizi 2018; Icarte et al. 2018)).

From Hierarchical RL approaches, it is known that abstractions play a fundamental role in subtask decomposition and efficient exploration. The technique proposed in this work allows to exploit abstractions of Markov Decision Processes (MDPs) to allow learning algorithms to effectively explore the *ground*[1] environment, while guaranteeing optimal convergence. The abstraction of some ground MDP $\mathcal{M}$ is an MDP $\mathcal{M}_\phi$ whose states represent sets of states of $\mathcal{M}$. A simple example is that of an agent moving in a map. States of $\mathcal{M}$ could determine the agent's position in terms of continuous coordinates, orientation, and other configurations. States of the abstraction $\mathcal{M}_\phi$, instead, may be coarser descriptions, for example, through discretization or by projecting out some state variables. Such compression corresponds, ultimately, to partitioning the concrete state-space and implicitly defines a mapping from concrete to abstract states. Importantly, action spaces of $\mathcal{M}$ and $\mathcal{M}_\phi$ may differ, as each model would include the actions that are best appropriate for each representation.

The core intuition is that, by first learning the optimal policy ρ_ϕ of the abstract MDP, we obtain a value estimate V^*_ϕ which can be exploited to guide learning on the ground model $\mathcal{M}$. Technically, we adopt a variant of Reward Shaping (RS), which is generated from V^*_ϕ, which offers rewards that are consistent with the correspondence between states at the ground and the abstract level. In this way, when learning in the concrete model $\mathcal{M}$, the agent is *biased* to visit first states corresponding to the abstract ones preferred by ρ_ϕ, thus trying, in a sense, to replicate the behavior of ρ_ϕ at the ground level.

For such exploration bias to be effective, it is essential that the transitions of $\mathcal{M}_\phi$ are good proxies for the dynamics of $\mathcal{M}$. We characterize this relation by identifying conditions under which the optimal policy of the ground MDP with computed rewards converges to a near-optimal exploration policy. We call such model the *biased MDP*.

An important difference with respect to previous works is that, since the proposed approach focuses on the definition of a novel RS mechanism, it is very general and may be com-

[1] We follow the nomenclature from (Li, Walsh, and Littman 2006).

bined with any off-policy learning algorithm. Furthermore, since computed rewards would not perturb the optimal policy, it's possible to iterate this process, giving raise to a linear hierarchy of abstract representations, each constituting a coarser model of the previous one. Again, one can take advantage of the experience obtained at the higher abstraction level to speed-up the learning process in the finer model.

The proposed approach is validated on several use cases, over which we also discuss properties and effectiveness of our reward shaping approach. The validation results integrate the theoretical results highlighting different cases with improved sample efficiency.

The contributions of this work include: (i) the definition of the multiple-level abstraction framework; (ii) the definition of a novel RS schema which allows for transferring the experience acquired at a coarser model to the finer one next in the abstraction hierarchy; (iii) the identification of a set of relations that allow to characterize near-optimal abstractions; (iv) the computation of an upper bound for the value loss of exploration, when using our RS (wrt optimal policy without RS) that depends on an explicit feature of the abstraction; (v) an empirical analysis showing that higher abstraction quality effectively improves sample-efficiency and that modelling errors yield only a limited performance degradation.

Preliminaries

Notation By $\Pi(\mathcal{Y})$ we denote the class of probability distributions over a set $\mathcal{Y}$. By $f : \mathcal{X} \to \Pi(\mathcal{Y})$, we denote a function returning a probability distribution (i.e., $f : \mathcal{X}, \mathcal{Y} \to [0, 1]$, with $\sum_{y \in \mathcal{Y}} f(x, y) = 1$, for all $x \in \mathcal{X}$).

Any total function $f : \mathcal{X} \to \mathcal{Y}$ induces a partition on its domain $\mathcal{X}$, s.t. two elements are in the same block iff $f(x) = f(x')$. We denote blocks of the partition by elements of $\mathcal{Y}$, thus writing $x \in y$ instead of $x \in \{x' \mid x' \in \mathcal{X}, f(x') = y\}$. When the partition refers to abstract states, blocks will also be referred to as abstract states.

Markov Decision Processes (MDPs) A Markov Decision Process (MDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $T : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ is the probabilistic transition function, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R} \subset \mathbb{R}$ is the reward function, and $0 \leq \gamma \leq 1$ is the discount factor. In the following, when the MDP is clear from the context, we write $T(s, a, s')$ and $p(s' \mid s, a)$ interchangeably.

The value of a policy ρ in some state s, denoted $V^\rho(s)$, is the expected sum of discounted rewards, when starting at s, and selecting actions based on ρ. Every MDP admits an optimal policy, $\rho^* = \arg\max_{\rho \in P} V^\rho(s)$, which is deterministic and Markovian (Puterman 1994). Thus, we consider policies as functions $\rho : \mathcal{S} \to \mathcal{A}$. The optimal value is abbreviated $V^* \equiv V^{\rho^*}$. The function $Q^\rho(s, a)$ is the value of executing $a \in \mathcal{A}$, then following policy ρ. For optimal actions, the two notions of value coincide $Q^*(s, \rho^*(s)) = V^*(s)$. Reinforcement Learning (RL) is the task of learning an optimal policy in an MDP with unknown T and R.

Reward Shaping (RS) Reward Shaping (RS) (Ng, Harada, and Russell 1999) is a technique for learning in MDPs with sparse rewards that occur rarely during exploration. The purpose of RS is to guide the agent by exploiting some prior knowledge in the form of additional rewards: $R^s(s, a, s') \coloneqq R(s, a, s') + F(s, a, s')$, where F is some *shaping* function. A fundamental requirement of RS is that the additional rewards should not modify the set of optimal policies. This is guaranteed by *Potential-Based RS* (Ng, Harada, and Russell 1999) (simply called "Reward Shaping" from now on) which adopts potential functions of the form:

$$F(s, a, s') \coloneqq \gamma\, \Phi(s') - \Phi(s) \tag{1}$$

or its variants, (Wiewiora, Cottrell, and Elkan 2003; Devlin and Kudenko 2012). In the infinite-horizon case, equation (1) guarantees that the set of optimal policies for $\mathcal{M}$ and $\mathcal{M}^s = \langle \mathcal{S}, \mathcal{A}, T, R^s, \gamma \rangle$ coincide, for any $\Phi : \mathcal{S} \to \mathbb{R}$.

Since optimal policies are not modified, RS only provides an initial bias to the learning algorithm; indeed, as shown by (Wiewiora 2003), the Q-learning algorithm over $\mathcal{M}^s$ performs the same updates as Q-learning over $\mathcal{M}$ with the modified Q-table initialization: $Q'_0(s, a) \coloneqq Q_0(s, a) + \Phi(s)$.

Options An option (Sutton, Precup, and Singh 1999), for an MDP $\mathcal{M}$, is a temporally-extended action, defined as $o = \langle \mathcal{I}_o, \rho_o, \beta_o \rangle$, where $\mathcal{I}_o \subseteq \mathcal{S}$ is an initiation set, $\rho_o : \mathcal{S} \to \Pi(\mathcal{A})$ is the policy to execute, and $\beta_o : \mathcal{S} \to \{0, 1\}$ is a termination condition that, from the last state, computes whether the option should terminate. Termination conditions sometimes follow a more general definition. Instead, the ones of interest in this paper are deterministic and Markovian. Adding options as possible actions to an MDP produces a model called semi-MDP. Similarly to MDPs, there always exist an optimal policy for a semi-MDP that only involves deterministic options. Thus, we only consider intra-option policy of the type $\rho_o : \mathcal{S} \to \mathcal{A}$.

ϕ-Relative Options (Abel et al. 2020) Given an MDP $\mathcal{M}$ and a mapping $\phi : \mathcal{S} \to \mathcal{S}_\phi$, an option $o = \langle \mathcal{I}_o, \rho_o, \beta_o \rangle$ of $\mathcal{M}$ is said to be *ϕ-relative* if and only if there is some $s_\phi \in \mathcal{S}_\phi$ such that, for all $s \in \mathcal{S}$:

$$\mathcal{I}_o = \{s \mid s \in s_\phi\}, \quad \beta_o(s) = \mathbb{I}(s \notin s_\phi), \quad \rho_o \in P_{s_\phi} \tag{2}$$

where $\mathbb{I}(\xi) = 1$ if ξ is true, 0 otherwise, and P_{s_ϕ} is the set of policies of the form $\rho_{s_\phi} : \{s \mid s \in s_\phi\} \to \mathcal{A}$ defined over states $s \in s_\phi$.

Framework

Consider an environment in which experience is costly to obtain. This might be a realistic simulation or an actual environment in which a physical robot is acting. This is our *ground* MDP $\mathcal{M}_0$, that we aim to solve while reducing the number of interactions with the environment. Instead of learning on this MDP directly, we choose to solve a simplified, related problem, that we call the *abstract* MDP. Once a solution of the abstract model is computed, such information can be used as a heuristic to speed up learning in the ground MDP. This idea is not limited to a single abstraction. Indeed, we consider a hierarchy of related MDPs $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_n$, of decreasing difficulty, where the experience acquired by an expert acting in $\mathcal{M}_i$ can be exploited to speed up learning in the previous one, $\mathcal{M}_{i-1}$.

Associated to each MDP abstraction $\mathcal{M}_i$, we also assume the existence of a *mapping* function $\phi_i : \mathcal{S}_i \to \mathcal{S}_{i+1}$, which maps states of $\mathcal{M}_i$ to states of its direct abstraction $\mathcal{M}_{i+1}$. Since each MDP in the sequence should be easier to solve with respect to its predecessor, we know that $|\mathcal{S}_i| \geq |\mathcal{S}_{i+1}|$. Therefore, the mapping induces a partition over $\mathcal{S}_i$, where each block contains all states that are mapped through ϕ_i to a single state in $\mathcal{M}_{i+1}$.

While our approach requires a mapping between states, it is important to highlight that we do not require any mapping between the action spaces, which will remain implicit. In other words, the agent solving the concrete model does not need to know which kind of actions have been used at the abstract level. Such minimal requirements leaves high flexibility to the designers in the definition of the abstraction hierarchy for a given problem.

An abstract model is therefore a suitable relaxation of the environment dynamics. For example, in a navigation scenario, an abstraction could contain actions that allow to just "leave the room", instead of navigating though space with lower-level controls. The mapping function in the state space naturally follows from the ground MDP and the chosen abstract state space, while no explicit mapping of the action spaces is required.

Exploiting the knowledge

Consider a hierarchy of abstractions $\mathcal{M}_0, \ldots, \mathcal{M}_n$ together with their state mapping functions $\phi_0, \ldots, \phi_{n-1}$. The learning process proceeds incrementally, training in order from the easiest to the hardest model. At each level, the knowledge acquired from an abstraction $\mathcal{M}_{i+1}$ can be used to speed up learning in the lower model $\mathcal{M}_i$. To do so, our method applies a form of Reward Shaping to $\mathcal{M}_i$, employing a potential that is derived from the abstraction solution.

In particular, we recognize that the optimal value function V^*_{i+1} can be used as an helpful advice that will evaluate how desirable a group of states is, according to the abstraction. Therefore, at each level instead of learning on $\mathcal{M}_i$, training is performed on a modified model, that we can call the *biased MDP*:

Definition 1. Let $\mathcal{M}_i$ be an MDP and $\mathcal{M}_{i+1}$ its abstraction, with associated mapping $\phi_i : \mathcal{S}_i \to \mathcal{S}_{i+1}$. We define the *biased MDP* of $\mathcal{M}_i$ with respect to $\mathcal{M}_{i+1}$, as the model $\mathcal{M}^b_i$, resulting from the application of reward shaping to $\mathcal{M}_i$, using the following potential:

$$\Phi(s) \coloneqq V^*_{i+1}(\phi_i(s)) \tag{3}$$

where, V^*_{i+1} is the optimal value function of the abstraction $\mathcal{M}_{i+1}$.

This choice for the RS is a novel contribution of this paper. With this choice, the potential of a state is evaluated according to how desirable is the corresponding state in the abstract model. This is beneficial, as high potentials are associated to high Q-function value initializations in the ground model (Wiewiora 2003).

Reward Shaping for Episodic RL

Potential-Based Reward Shaping has been explicitly designed not to alter the optimal policies. In fact, regardless of the potential Φ, in case of an infinite horizon, or if episodes always terminate in a zero-potential absorbing state, this is always guaranteed (Ng, Harada, and Russell 1999). However, in RL, we often diversify agent experiences by breaking up exploration in episodes of finite length. Thus, in the episodic setting, these guarantees do not hold anymore, as time limits might terminate episodes in states with arbitrary potential. As a consequence, the optimal policy may be altered (Grzes 2017).

To see this, consider an episode $\pi = s_0 a_1 r_1 s_1 \ldots s_n$ of an MDP $\mathcal{M}$, and the associated episode $\pi' = s_0 a_1 r'_1 s_1 \ldots s_n$, where rewards altered via reward shaping. The returns of the two episodes are related by (Grzes 2017):

$$\begin{aligned} G(\pi') &\coloneqq \sum_{t=0}^{n} \gamma^t \, r'_{t+1} \\ &= G(\pi) + \gamma^n \, \Phi(s_n) - \Phi(s_0) \end{aligned} \tag{4}$$

where $G(\pi)$ is the original return computed from $\mathcal{M}$, and $\Phi(s_0)$ is constant with respect to the actions. Since $\gamma^n \, \Phi(s_n)$ is the term which can perturb optimal policies, the solution proposed by (Grzes 2017) is to assume, for every terminal state, the null potential $\Phi(s_n) = 0$, as this would preserve the original returns and policies.

However, this is not always the only desirable solution. In fact, we might be interested in relaxing the guarantees of convergence to an identical policy, in favour of a stronger impact on learning speed. The same need has been also identified by (Schubert, Oguz, and Toussaint 2021).

As an example, let us consider an MDP with a null reward function everywhere, except when transitioning to a distinct goal state. Regardless of the potential, as a consequence of equation (4), all finite trajectories which do not contain the goal state are associated to the same return. Since the agent cannot estimate its distance to the goal state through a difference in returns, return-invariant RS of (Grzes 2017) is not able to provide a persistent exploration bias to the agent.

The RS adopted in this paper, which is formulated in Definition 1, does not assign null potentials to terminal states. Therefore, we say that it is *not* return-invariant. This explains why the MDP of Definition 1 has been called "biased": optimal policies of $\mathcal{M}^b_i$ and $\mathcal{M}_i$ do not necessary correspond.

Optimal policy learning

Since we deliberately adopt a form of RS which is not return invariant, we devised a technique to recover the original optimality guarantees. The present section proves that the proposed method still converges to the optimal policy, when Q-learning (Watkins and Dayan 1992) is used. However, the same result also apply to other RL algorithms that share the same convergence requirements of Q-learning, as most off-policy algorithms do.

Our method proceeds as follows. When learning on MDP $\mathcal{M}_i$, we perform updates on two distinct Q-function estimates:

$\hat{Q}^{b*}_i$ called *active function*, is in charge of estimating Q^{b*}_i, that is the optimal Q-value function of the biased MDP $\mathcal{M}^b_i$.

$\hat{Q}_i^*$ called *passive function*, is an estimate for Q_i^*, the optimal Q-value function of the original $\mathcal{M}_i$.

The exploration policy used to collect experience is computed from the current active function estimate (hence the name "active"). Therefore, the learning algorithm proceeds normally with respect to $\hat{Q}^{b*}$. The passive function, instead, is updated from the same state-action transitions as those produced for training the active function, with the only difference that rewards are generated from the original reward function R_i, without reward shaping applied.

Clearly, it is essential that each of those estimates can converge to the desired quantity. Thus, the classic Q-learning convergence assumptions are required:

Assumption 1. Actions are selected according to a stochastic exploration policy $\rho_e : \mathcal{S} \to \Pi(\mathcal{A})$, that satisfies:

$$\rho_e(s,a) > 0 \qquad \forall s \in \mathcal{S}_i,\, \forall a \in \mathcal{A}_i \tag{5}$$

Assumption 2. The sequence of learning rates $(\alpha_t)_{t\in\mathbb{N}}$ used in the Q-learning update rule satisfies:

$$\sum_{t=0}^{\infty} \alpha_t = \infty \qquad \sum_{t=0}^{\infty} \alpha_t^2 < \infty \tag{6}$$

As a consequence, both functions converge to the desired values:

Proposition 1. *Let $\hat{Q}_i^{b*}$ and $\hat{Q}_i^*$ be the Q-value estimations for $\mathcal{M}_i^b$ and $\mathcal{M}_i$, updated according to active-passive Q-learning described above. Then, under assumptions 1 and 2, both $\hat{Q}_i^{b*}$ and $\hat{Q}_i^*$ converge to Q_i^{b*} and Q_i^*, respectively.*

Proof. Assumption 1 on the exploration policy ρ_e ensures that, in every state of $\mathcal{M}_i^b$, each action is executed infinitely often. Together with assumption 2, they guarantee that $\hat{Q}_i^{b*}$, under Q-learning updates, converges to Q_i^{b*}. The same sequence of state-action transitions (s_t, a_{t+1}, s_{t+1}) is also used for the Q-learning update on $\hat{Q}_i^*$. Since $\mathcal{M}_i$ and $\mathcal{M}_i^b$ share the same state and action spaces, such sequence is consistent with the same exploration policy ρ_e, when executed on $\mathcal{M}_i$, which satisfies assumption 1. Thus, also $\hat{Q}_i^*$ converges to Q_i^*, regardless of rewards ($\mathcal{M}_i^b$ rewards are also bounded). □

In particular, a decaying ϵ-greedy policy computed from $\hat{Q}_i^{b*}$ is a possible exploration policy that satisfies assumption 1. In fact, among all exploration policies, we're interested in those, whose deterministic actions depend on the active function's estimate, as those incorporate the useful exploration biases coming from the abstraction. Although the passive function is updated from the same experience, it maintains an estimate for the original MDP and it converges to its optimal Q-function Q_i^*.

At convergence, the passive Q-function can be used, in turn, to compute the potential needed to drive the next abstraction level, or, when $\mathcal{M}_i$ is the target ground model $\mathcal{M}_0$, it constitutes the final learning objective.

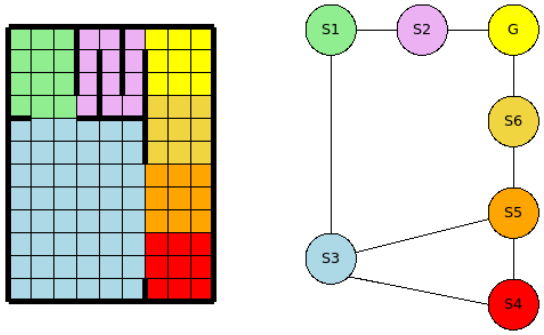


Figure 1: Motivating example: ground MDP (left) and abstract MDP (right).

Abstraction quality

Due to how the framework is designed, convergence on a model $\mathcal{M}_i$, does depend on its abstraction, $\mathcal{M}_{i+1}$, but not on any other model. Therefore, when discussing convergence properties, it suffices to talk about a generic ground MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ and its direct abstraction, $\mathcal{M}_\phi = \langle \mathcal{S}_\phi, \mathcal{A}_\phi, T_\phi, R_\phi, \gamma_\phi \rangle$, while ignoring the rest of the hierarchy. Also, let $\phi : \mathcal{S} \to \mathcal{S}_\phi$ denote the relevant mapping. An example of abstraction between two MDPs is graphically shown in Figure 1, where the different colors indicate how the mapping function transforms each group of cells to a state of the abstract MDP.

Given a ground model $\mathcal{M}$, different abstractions induce different biases in the exploration policy. Some abstractions may even slow down convergence by encouraging exploration toward irrelevant portions of the state space. This section serves to describe what properties should a *good* abstraction possess.

We start our discussion from the following intuitive observations:

- Every abstract state $s_\phi \in \mathcal{S}_\phi$ corresponds to a *set of states* $\phi^{-1}(s_\phi) \subseteq \mathcal{S}$, in the ground MDP.
- Every abstract action $a_\phi \in \mathcal{A}_\phi$ corresponds to a *partial policy* in the ground MDP.

In particular, abstract actions should be associated to non-interruptible policies that terminate when leaving the current block, when a new abstract state is reached. So, a more appropriate correspondence can be identified between actions $\mathcal{A}_\phi$ in $\mathcal{M}_\phi$ and ϕ-relative options in $\mathcal{M}$.

Goal MDPs Although we may apply the proposed method in generic MDPs, Definition 1 of the biased model makes it specifically effective in a class of problems that can be modelled as Goal MDPs:

Definition 2. We say that an MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ is a *Goal MDP* iff there exists a set of goal states $\mathcal{G} \subseteq \mathcal{S}$ such that:

$$R(s,a,s') = \begin{cases} 1 & \text{if } s \notin \mathcal{G} \text{ and } s' \in \mathcal{G} \\ 0 & \text{else} \end{cases} \tag{7}$$

$$V^*(s) = 0 \qquad \forall s \in \mathcal{G} \tag{8}$$

Equivalent definitions of Goal MDPs often assume goals to be absorbing states. Equation (8) simply requires that from any goal, it is not possible to re-enter any goal state and collect an additional reward. Since we are not interested in the agent's behaviour at goal states, episodes may be terminated once the agent reaches some $s_n \in \mathcal{G}$.

Goal MDPs capture many interesting real-world problems, in which the agent is rewarded upon successful completion of a task (the MDP may also contain dead-ends associated to failures). In the following discussion, we focus on this class of problems:

Assumption 3. The ground MDP $\mathcal{M}$ is a Goal MDP.

Given the above assumption, each abstraction should be also consistent with this set of ground goal states:

Assumption 4. Given a Goal MDP $\mathcal{M}$ with goal states $\mathcal{G}$, its abstraction, $\mathcal{M}_\phi$, is a Goal MDP with goal states $\mathcal{G}_\phi$ satisfying:

$$\mathcal{G} = \cup_{s_\phi \in \mathcal{G}_\phi} \phi^{-1}(s_\phi) \tag{9}$$

Since $\phi : \mathcal{S} \to \mathcal{S}_\phi$ induces a partition on $\mathcal{S}$, assumption 4 implies that goal states in the abstraction must correspond to all and only those states that are goals in the ground model:

$$\forall s, s' \in \mathcal{S} : \; (s \in \mathcal{G}) \wedge (s' \notin \mathcal{G}) \Rightarrow \phi(s) \neq \phi(s') \tag{10}$$

$$\mathcal{G}_\phi = \{\phi(s) \mid s \in \mathcal{G}\} \tag{11}$$

In the example, yellow cells in the ground MDP mapped to abstract state `G` represent the goal and all other cells of a different color are not goal states.

The main objective of this section is to quantify, and provide a bound for, the difference between the optimal policy of the original problem and the optimal policy of the biased MDP, as induced by the abstraction. By showing conditions under which these two policies achieve similar value, we ensure that the agent's exploration policy will converge to an optimum. We establish this in Theorem 1, by exploiting the options' multi-step transition model.

Multi-step transition model Relaxing the usual notation, $Q^*(s, a)$, we denote with $Q^*(s, o)$ the expected return of executing the option o, and following the optimal policy afterwards. Similarly, $Q(s, \rho)$ denotes the expected return of executing the policy ρ until the end of the episode.

By combining the classic multi-step return of options (Sutton, Precup, and Singh 1999), ϕ-relative options from (Abel et al. 2020) and Goal MDPs of Definition 2, we obtain:

$$\begin{aligned} Q^*(s,o) &\coloneqq \mathbb{E}_{s'|s,o}[R(s,\rho_o(s),s') + \gamma(\\ &\qquad \mathbb{I}(s' \in s_\phi)\, Q^*(s',o) + \mathbb{I}(s' \notin s_\phi)\, V^*(s'))] \\ &= \sum_{k=0}^{\infty} \gamma^k \sum_{s_{1:k} \in \phi(s)^k} \sum_{s' \notin \phi(s)} (\\ &\qquad p(s', s_{1:k} \mid s, \rho_o) \left(\mathbb{I}(s' \in G) + \gamma\, V^*(s')\right)\Big) \end{aligned} \tag{12}$$

where $\sum_{s_{1:k} \in \phi(s)^k}$ is a sum over all possible sequences $s_{1:k} \coloneqq s_1 \dots s_k$ of k states that remain within $\phi(s)$.

Equation (12) is not an expression about abstract states, because it's still relevant which ground state s' is reached at the end of the option. In the following definition, we introduce a parameter δ that quantifies how much these states s' are dissimilar in value. This allows to jointly talk about the value of the group of states that can be reached with each option. Therefore, we define a function $W_\delta : \mathcal{S}_\phi \times \mathcal{S}_\phi \to \mathbb{R}$, that, given a pair of abstract states (s_ϕ, s'_ϕ), predicts, with δ-approximation error, the approximate value of the successor ground states $s' \in s'_\phi$ (unknown at the time of the prediction) that can be reached from $s \in s_\phi$.

Definition 3. Consider an MDP $\mathcal{M}$, abstraction $\mathcal{M}_\phi$, and mapping ϕ. We define the *abstract value approximation* as the smallest $\delta \geq 0$ such that there exists a function $W_\delta : \mathcal{S}_\phi \times \mathcal{S}_\phi \to \mathbb{R}$, that satisfies:

$$\begin{aligned} &\forall s \in s_\phi,\, \forall s' \in s'_\phi,\, \forall a \in \mathcal{A} \\ &\qquad p(s' \mid s, a) > 0 \Rightarrow |W_\delta(s_\phi, s'_\phi) - V^*(s')| \leq \delta \end{aligned} \tag{13}$$

Thus, according to this definition, the frontier separating any two sets in the partition lies in states that are equally valuable with respect to the goal, with an approximation error of at most δ. For example, in Figure 1, ground states between two neighbouring regions would be approximated to be equivalently close to the final destination in yellow. On the other hand, a given δ parameter poses a constraint on the mapping function.

Thanks to definition 3, it is possible to compute a bound for the value of options, only taking future abstract states into consideration. For this purpose, in the following, we use $p(s'_\phi, k)$ to denote the probability of the event of remaining for k steps in the same abstract state, then reaching block s'_ϕ at the next transition.

Lemma 1. *Let $\mathcal{M}$ be a Goal MDP and $\mathcal{M}_\phi$ its abstraction according to a mapping function ϕ, such that they satisfy assumptions 3 and 4. The value of an option in $\mathcal{M}$ admits the following lower bound:*

$$\begin{aligned} Q^*(s,o) \geq \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k\, p(s'_\phi, k \mid s, \rho_o) \big(\\ \mathbb{I}(s'_\phi \in G) + \gamma\, (W(\phi(s), s'_\phi) - \delta)\big) \end{aligned} \tag{14}$$

where, δ and W_δ follow definition 3.

Proof. Proof in appendix. □

This lemma provides a different characterization of options, in terms of abstract states, so that it can be exploited to obtain the theorem below.

Limiting the suboptimal exploration The section "optimal policy learning" has shown that the experience generated while learning on the biased MDP $\mathcal{M}^b$ is collected and used to learn the optimal policy for $\mathcal{M}$. Since convergence is guaranteed, we know that this exploration policy used for learning converges to ρ^{b*}. Thus, we are interested in limiting the difference, in value, between ρ^{b*} and ρ^*.

Definition 4. ($\mathcal{M}_\phi$-value loss) The value loss of $\mathcal{M}_\phi$, abstraction of an MDP $\mathcal{M}$, is the expected value loss of executing ρ^{b*}, the optimal policy of $\mathcal{M}^b$, on $\mathcal{M}$, instead of its optimal policy:

$$L(\mathcal{M}, \mathcal{M}_\phi) \coloneqq \max_{s \in \mathcal{S}} |V^*(s) - Q(s, \rho^{b*})| \tag{15}$$

We expand the results from (Abel et al. 2020) to limit the above loss. Thanks to Lemma 1, we know that the value of ρ^{b*}, only depends on the k-step transition probability to each abstract state. So, we assume this quantity is bounded:

Assumption 5. Let ρ^* and ρ^{b*} be the optimal policies for the original $\mathcal{M}$ and biased MDP $\mathcal{M}^b$, respectively. We assume that there exists $\epsilon \geq 0$ such that:

$$\forall s \in \mathcal{S}, \quad \forall s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}, \quad \forall k \in \mathbb{N}$$
$$|\, p(s'_\phi, k \mid s, \rho^*) - p(s'_\phi, k \mid s, \rho^{b*}) \,| \leq \epsilon \tag{16}$$

It is now finally possible to state:

Theorem 1. *Let $\mathcal{M}$ be an MDP and $\mathcal{M}_\phi$ its abstraction, satisfying assumptions 3, 4 and 5. The value loss of $\mathcal{M}_\phi$ satisfies:*

$$L(\mathcal{M}, \mathcal{M}_\phi) \leq \frac{\epsilon + \gamma\, |\mathcal{S}_\phi|\, (\epsilon + \epsilon\, \delta + 2\, \delta))}{(1-\gamma)^2} \tag{17}$$

Proof. Proof in appendix. □

For a $\delta = 0$, this bound has similarities with the inequality n. 5 in (Abel et al. 2020). Notice however, that the one stated here is expressed in terms of the size of the abstract state space, which usually can be assumed to be $|\mathcal{S}_\phi| \ll |\mathcal{S}|$.

Theorem 1 provides an important contribution of this paper showing that, when acting according to biases derived from the abstraction, the agents achieves near optimal value on the original domain. This characterizes abstractions that induce good biases onto the lower model. However, we recall that convergence is guaranteed regardless of the abstraction quality, as the agent can explore from an ϵ-greedy policy built from ρ^{b*}.

Validation

Navigation task

We initially consider a navigation scenario, where an agent has to navigate through a map and reach a desired location.

Environment The ground MDP $\mathcal{M}_1$ consists of a finite state space $\mathcal{S}_1$, containing a set of locations, and actions $\mathcal{A}_1$ that allow to move between neighbour locations, with some small failure probability. Figure 2 contains an example of such domains that we will use for training. The gray cells are goal states. So, R_1 generates a positive reward upon entering in this right-most region.

As abstraction for this environment, we define an MDP $\mathcal{M}_2$, with $\mathcal{S}_2 = \{s_0, \ldots, s_7\}$, where each of these states represents one of the colored regions. The mapping $\phi_1 : \mathcal{S}_1 \to \mathcal{S}_2$ is defined according to this choice. Actions $\mathcal{A}_2$ allow to move, with high probability, from any region to any other, only if they are connected in $\mathcal{M}_1$.

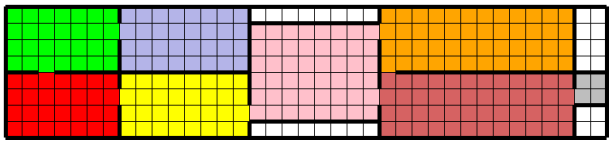
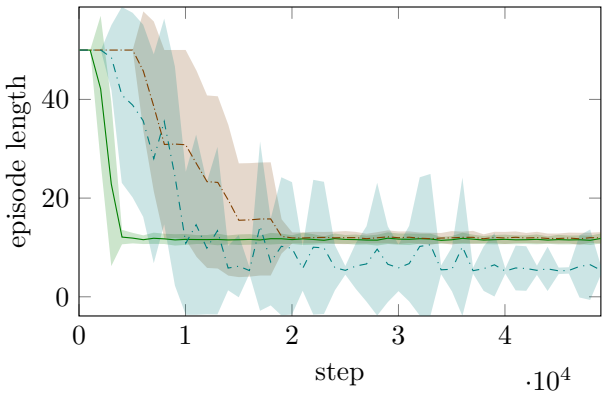
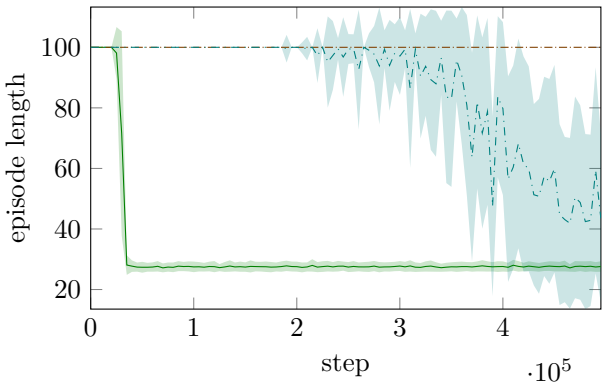


Figure 2: A grid-world domain for the navigation task.



(a) Navigation task in 4-rooms domain (Abel et al. 2020).



(b) Navigation task in 8-rooms domain of Figure 2.

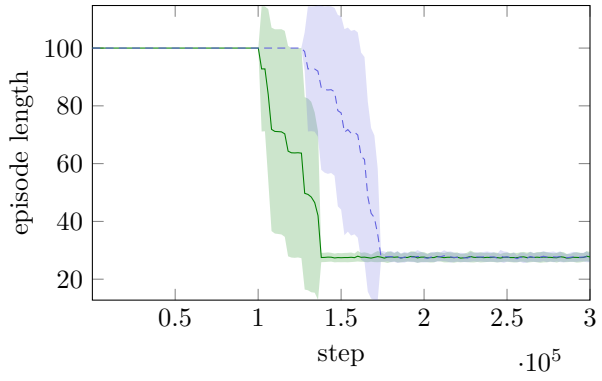
Figure 3: Results on the navigation tasks.

Training results In the plots of figures 3a, 3b, we compare performances of the following algorithms:

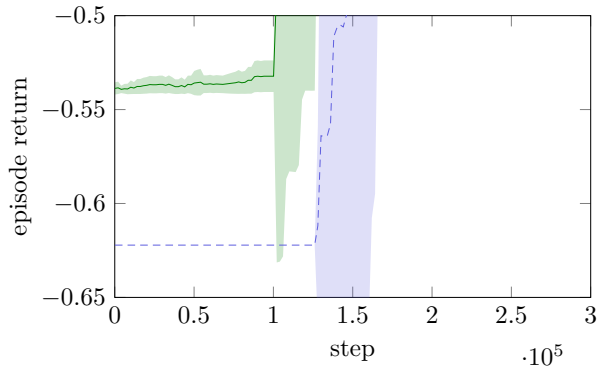
- Q-learning without reward shaping;
- Delayed Q-learning (Strehl et al. 2006);
- Q-learning with rewards computed as in Definition 1. Evaluation of the passive Q-function.

Episodes are terminated after a fixed timeout or when the agent reaches a goal state. Therefore, lower episode lengths are associated to higher cumulative returns. Each point in the plot is an average of 10 different runs with 10 episodes each. Shaded areas are the standard deviations. Additional training details can be found in the appendix.

As we can see from Figure 3a, all algorithms, even under Q-learning naive exploration policy converge to the optimum relatively fast on the 4-rooms domain. In Figure 3b, as the state spaces starts to increase and to be harder to explore,



(a)



(b)

Figure 4: Comparison of different forms of reward shaping. Q-learning with ϵ-greedy 1-to-0 linear decay.

Delayed Q-learning still manages to converge. However, our agent in green, the one driven by computed rewards, steadily reaches the optimum in less than 500 episodes. Instead, the same Q-learning algorithm, without any advice cannot converge in reasonable time.

Return-invariant shaping

As discussed in previous sections, when applying RL for the episodic setting, there is a delicate distinction to make:

- Return-invariant RS, which assigns null potentials at terminal states.
- Non return-invariant RS (our approach).

Figure 4a compares both variants on the same environment of Figure 2. As we can see, even though the learning algorithm is the same, our approach gives exploration biases in a more effective way. The reason for this behaviour can be seen by looking at the cumulative returns that the two agent accumulate before reaching the goal even once. The two are reported in Figure 4b. Although the two returns are incomparable (as they receive different rewards), the left half of the plot shows that the return-invariant agent always achieves the null return, regardless of how close it might be to the goal. The variability on the green line, instead, shows

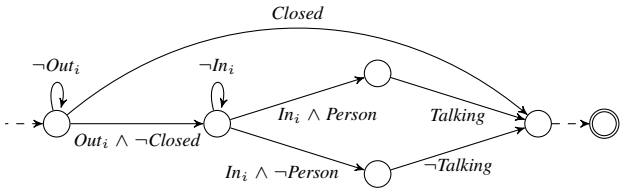


Figure 5: A temporally-extended task in the navigation domain. Arcs are associated to environment events.

that the agent is rewarded differently, depending on the value of each abstract state reached.

Interaction task

In this section, we demonstrate that the proposed method applies to a wide range of algorithms and environments. With respect to variability in environments, we emphasize that Goal MDPs can capture many interesting problems, apart from those considered up to this point. For this purpose, let us consider the same navigation scenario, but instead of reaching a location, the task assigned to the robot is a temporally-extended behavior such as: "reach the entrance of each room and, if the door is open, enter and interact with the person inside, if present". This task is summarized by the deterministic automaton $\mathcal{A}$ of Figure 5. Note that there is a single accepting state.

Regarding the environment dynamics, instead, apart from the abstract and grid representations that we've seen in the previous case, the robot movements are now also modelled at a much lower resolution, using continuous features. For this purpose, let us consider an MDP $\mathcal{M}_{0,d}$, in which states (x, y, θ) represent the agent's mobile base position and orientation in the plane. Actions are relative displacements or rotations with respect to the current configuration.

There exists a Goal MDP $\mathcal{M}_0$ that captures both the dynamics and the task defined above, which can be obtained through a suitable composition of $\mathcal{M}_{0,d}$ and $\mathcal{A}$. Such reduction has been described in detail in (Brafman, De Giacomo, and Patrizi 2018; Icarte et al. 2018). Abstractions $\mathcal{M}_1$ and $\mathcal{M}_2$ also follow a similar composition.

Since the MDP $\mathcal{M}_0$ includes continuous features, we now adopt a Deep RL algorithm, namely Dueling DQN (Wang et al. 2016). The plot in Figure 6 shows a training comparison between the Dueling DQN agent alone (dot-dashed brown), and Dueling DQN with computed rewards from the grid abstraction. As we can see, our method allows to provide useful exploration bias even when using diverse learning algorithms and features.

Robustness to modelling errors

Finally, we analyzed the training performance in presence of significant modelling errors in the abstraction. For this test we train on the same navigation task of Figure 2, used in previous experiments. Now, the same Q-learning agent is trained from rewards constructed from three different abstract MDPs:

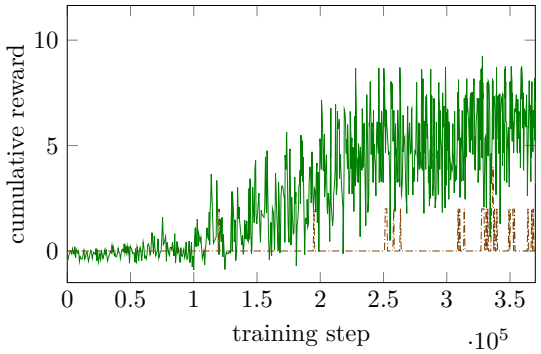


Figure 6: Dueling DQN algorithm with decaying ϵ-greedy exploration. Goal reward is 10. Average of 5 runs. The y-axis shows the cumulative rewards obtained during training.

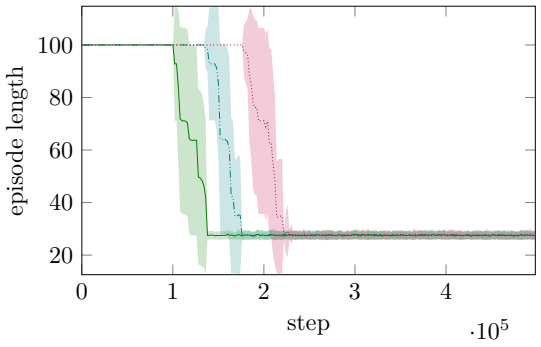


Figure 7: Episode lengths per training episode, when reward shaping is computed in presence of modelling errors.

- $\mathcal{M}_2$: the same abstraction used in previous tests; two abstract states are connected if two colored regions are directly connected.
- $\mathcal{M}_2^{(b)}$: same abstraction as $\mathcal{M}_2$ with an additional transition from the central pink states to the goal states (not achievable in $\mathcal{M}_1$).
- $\mathcal{M}_2^{(c)}$: same abstraction $\mathcal{M}_2^{(b)}$ with an additional transition from the blue cells to the central pink region (again not achievable in $\mathcal{M}_1$).

Results are shown in Figure 7, which show evaluation performances among 10 different runs over 10 episodes each. Clearly, abstractions with bigger differences with respect to the underlying domain cause the learning process to slow down. However, with any of these abstraction, Q-learning converges to the desired policy and performance in terms of sample efficiency degrade gracefully. Interestingly, even in presence of severe modelling errors, the abstraction still provides useful information with respect to uninformed exploration.

Related work

The field of Hierarchical RL specifically studies efficiency of algorithms in presence of abstractions. Some classic approaches are MAXQ (Dietterich 2000), HAM (Parr and Russell 1998) and the Options framework (Sutton, Precup, and Singh 1999). In order to describe which relation should the ground MDP and its abstraction satisfy, (Ravindran and Barto 2002; Li, Walsh, and Littman 2006) develop MDP Homomorphisms and approximated extensions. Differently to these works, our method does not try to capture symmetries and spatial regularities in the domain, rather, we are interested in coarse partitioning of neighbouring states, whose set can hardly be approximated with a single optimal value. Our abstractions are more closely related with those described in (Abel, Hershkowitz, and Littman 2016; Abel et al. 2020), which admit to talk about abstractions both with respect to state and action spaces. Still, they do not exploit, as in our work, explicit abstract MDPs, since they only learn on one ground model, from a given set of options.

Regarding the use of Reward Shaping, (Gao and Toni 2015) presented the idea of applying RS in context of HRL, and applying it specifically to the MAXQ algorithm. Our method, instead, can be combined with various learning algorithms. Recently, (Schubert, Oguz, and Toussaint 2021) proposed an effective form of biased RS for Goal MDPs. Differently to this work, our approach maintains the original optimality guarantees.

Another set of related works (Marthi; Grześ and Kudenko 2008; Biza and Jr. 2019; Steccanella, Totaro, and Jonsson 2021), with a different objective with respect to this paper, consider how abstractions may be learnt instead of being pre-defined. This is an interesting future extension for our method as well.

This work considers abstract models being MDPs. Planning models can be also regarded as abstractions, whose action schemas correspond to high-level actions (Illanes et al. 2020; Lee et al. 2021). Planning domains allow to extract useful heuristics which may also be goal-independent (Gehring et al. 2021). MDP abstractions, one the other hand, can be stochastic and only a simulator is required.

Conclusion

In this paper, we have presented an approach to increase sample-efficiency in RL, based on a linear hierarchy of abstract simulators and a form of reward shaping. While the ground $\mathcal{M}$ accurately captures the environment dynamics, higher-level models represent increasingly coarser abstractions of it. We have devised the theoretical framework for the approach and performed a validation activity. The results show effectiveness of the approach: we can take advantage of the policy learned at some level to speed-up the learning process on the lower-level. Importantly, our approach is completely general, as it makes little assumptions on the learning algorithm used and it has minimal requirements in terms of mapping between the abstraction layers.

For future work, we plan to develop a more comprehensive experimental analysis, comparing the proposed method with other approaches, mostly from hierarchical RL, and different forms of reward shaping. We also aim at demonstrating this framework in the context of robotic applications in which the most concrete MDP is a real robot acting in a complex environment through low-level perception and controls.

Acknowledgments

This work is partially supported by the ERC Advanced Grant WhiteMech (No. 834228), by the EU ICT-48 2020 project TAILOR (No. 952215), by the PRIN project RIPER (No. 20203FFYLK), and by the JPMorgan AI Faculty Research Award "Resilience-based Generalized Planning and Strategic Reasoning".

References

Abel, D.; Hershkowitz, D.; and Littman, M. 2016. Near Optimal Behavior via Approximate State Abstraction. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, 2915–2923. PMLR.

Abel, D.; Umbanhowar, N.; Khetarpal, K.; Arumugam, D.; Precup, D.; and Littman, M. 2020. Value Preserving State-Action Abstractions. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, 1639–1650. PMLR.

Biza, O.; and Jr., R. P. 2019. Online Abstraction with MDP Homomorphisms for Deep Learning. In Elkind, E.; Veloso, M.; Agmon, N.; and Taylor, M. E., eds., *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, 1125–1133. International Foundation for Autonomous Agents and Multiagent Systems.

Brafman, R. I.; De Giacomo, G.; and Patrizi, F. 2018. LTLf/LDLf Non-Markovian Rewards. In *AAAI*, 1771–1778.

Devlin, S.; and Kudenko, D. 2012. Dynamic Potential-Based Reward Shaping. In *AAMAS*, 433–440. Richland, SC.

Dietterich, T. G. 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition. *JAIR*, 13: 227–303.

Gao, Y.; and Toni, F. 2015. Potential based reward shaping for hierarchical reinforcement learning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Gehring, C.; Asai, M.; Chitnis, R.; Silver, T.; Kaelbling, L. P.; Sohrabi, S.; and Katz, M. 2021. Reinforcement Learning for Classical Planning: Viewing Heuristics as Dense Reward Generators. abs/2109.14830.

Grzes, M. 2017. Reward Shaping in Episodic Reinforcement Learning. In *AAMAS*, 565–573.

Grześ, M.; and Kudenko, D. 2008. Multigrid reinforcement learning with reward shaping. In *ICANN*, 357–366.

Icarte, R. T.; Klassen, T.; Valenzano, R.; and McIlraith, S. 2018. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *ICML*, 2107–2116.

Illanes, L.; Yan, X.; Icarte, R. T.; and McIlraith, S. A. 2020. Symbolic Plans as High-Level Instructions for Reinforcement Learning. In *ICAPS*, 540–550. AAAI Press.

Lee, J.; Katz, M.; Agravante, D. J.; Liu, M.; Klinger, T.; Campbell, M.; Sohrabi, S.; and Tesauro, G. 2021. AI Planning Annotation in Reinforcement Learning: Options and Beyond. *Planning and Reinforcement Learning PRL Workshop at ICAPS*.

Li, L.; Walsh, T. J.; and Littman, M. L. 2006. Towards a Unified Theory of State Abstraction for MDPs. In *ISAIM*, 531–539.

Marthi, B. ???? Automatic shaping and decomposition of reward functions. In *(ICML*.

Ng, A. Y.; Harada, D.; and Russell, S. J. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *ICML*, 278–287. San Francisco, CA, USA.

Parr, R.; and Russell, S. 1998. Reinforcement learning with hierarchies of machines. *Advances in neural information processing systems*, 1043–1049.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley.

Ravindran, B.; and Barto, A. G. 2002. Model Minimization in Hierarchical Reinforcement Learning. In *Abstraction, Reformulation and Approximation, 5th International Symposium, SARA 2002*, volume 2371 of *Lecture Notes in Computer Science*, 196–211. Springer.

Schubert, I.; Oguz, O. S.; and Toussaint, M. 2021. Plan-based relaxed reward shaping for goal-directed tasks. In *9th international conference on learning representations, ICLR 2021, virtual event, austria, may 3-7, 2021*. OpenReview.net.

Steccanella, L.; Totaro, S.; and Jonsson, A. 2021. Hierarchical Representation Learning for Markov Decision Processes. *CoRR*, abs/2106.01655.

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

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1-2): 181–211.

Wang, Z.; Schaul, T.; Hessel, M.; van Hasselt, H.; Lanctot, M.; and de Freitas, N. 2016. Dueling Network Architectures for Deep Reinforcement Learning. In *ICML*, volume 48, 1995–2003. JMLR.org.

Watkins, C. J.; and Dayan, P. 1992. Q-learning. *Machine learning*, 8(3-4): 279–292.

Wiewiora, E. 2003. Potential-based shaping and Q-value initialization are equivalent. *JAIR*, 19: 205–208.

Wiewiora, E.; Cottrell, G. W.; and Elkan, C. 2003. Principled methods for advising reinforcement learning agents. In *ICML*, 792–799.

Appendix

Lemma 1

It is possible to marginalize the probabilities in equation (12) over all possible sequences of states $s_{1:k}$. For this purpose, let us write $p(s', k)$ to represent the probability of the event of remaining for k steps in the same abstract state, then reaching s' at the next transition. We can now rephrase (12) into:

$$Q^*(s,o) = \sum_{s' \in \mathcal{S} \setminus \phi(s)} \sum_{k=0}^{\infty} \gamma^k \, p(s', k \mid s, \rho_o) \left(\mathbb{I}(s' \in G) + \gamma \, V^*(s') \right) \tag{18}$$

For all states in $\phi(s')$ that can be reached in one transition from $\phi(s)$, definition 3 applies, and we can introduce suitable $W : \mathcal{S}_\phi \times \mathcal{S}_\phi \to \mathbb{R}$ and δ, such that it is possible to provide the lower bound $W(\phi(s), \phi(s')) - \delta$ for each term $V^*(s')$, in the above sum. Therefore, we get:

$$\begin{aligned} Q^*(s,o) \geq \sum_{s' \in \mathcal{S} \setminus \phi(s)} \sum_{k=0}^{\infty} \gamma^k \, p(s', k \mid s, \rho_o) \, \big(\\ \mathbb{I}(s' \in G) + \gamma \, (W(\phi(s), \phi(s')) - \delta) \big) \end{aligned} \tag{19}$$

It is now possible to split the sum $\sum_{s' \in \mathcal{S} \setminus \phi(s)}$ into $|\mathcal{S}_\phi|$ sums over future blocks and marginalize them to obtain:

$$\begin{aligned} Q^*(s,o) \geq \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \, p(s'_\phi, k \mid s, \rho_o) \, \big(\\ \mathbb{I}(s'_\phi \in G) + \gamma \, (W(\phi(s), s'_\phi) - \delta) \big) \end{aligned} \tag{20}$$

This proves the lemma. A similar result holds for the upper constructed from $W(\phi(s), s'_\phi) + \delta$.

Theorem 1

Consider ρ^*, the optimal policy for $\mathcal{M}$, and ρ^{b*}, the optimal policy for the biased MDP. The value loss of $\mathcal{M}_\phi$ is defined as:

$$L(\mathcal{M}, \mathcal{M}_\phi) \coloneqq \max_{s \in \mathcal{S}} |V^*(s) - Q(s, \rho^{b*})| \tag{21}$$

Instead of computing such loss for the whole policies, we compute the loss of executing each initial portion: namely, the value of the associate ϕ-relative options constructed from them, and optimal policies afterwards.

$$|Q^*(s,o^*) \; - \; Q^*(s,o^{b*})| = Q^*(s,o^*) - Q^*(s,o^{b*}) \tag{22}$$

$$\begin{aligned} = \sum_{s' \in \mathcal{S} \setminus \phi(s)} \sum_{k=0}^{\infty} \gamma^k \, p(s', k \mid s, \rho_{o^*}) \left(\mathbb{I}(s' \in G) + \gamma \, V^*(s') \right) - \\ \sum_{s' \in \mathcal{S} \setminus \phi(s)} \sum_{k=0}^{\infty} \gamma^k \, p(s', k \mid s, \rho_{o^{b*}}) \left(\mathbb{I}(s' \in G) + \gamma \, V^*(s') \right) \end{aligned} \tag{23}$$

It is now possible to apply lemma 1 to both terms and bound the above loss,

$$|Q^*(s,o^*) \; - \; Q^*(s,o^{b*})| \tag{24}$$

$$\leq \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \, p(s'_\phi, k \mid s, \rho_{o^*}) \, \big(\tag{25}$$

$$\mathbb{I}(s'_\phi \in G) + \gamma \, (W(\phi(s), s'_\phi) + \delta) \big) -$$

$$\sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \, p(s'_\phi, k \mid s, \rho_{o^{b*}}) \, \big(\tag{26}$$

$$\mathbb{I}(s'_\phi \in G) + \gamma \, (W(\phi(s), s'_\phi) - \delta) \big) \tag{27}$$

$$= \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \left(p(s'_\phi, k \mid s, \rho_{o^*}) - p(s'_\phi, k \mid s, \rho_{o^{b*}}) \right) \big(\tag{28}$$

$$\mathbb{I}(s'_\phi \in G) + \gamma \, W(\phi(s), s'_\phi) \big) +$$

$$\sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \left(p(s'_\phi, k \mid s, \rho_{o^*}) + p(s'_\phi, k \mid s, \rho_{o^{b*}}) \right) \gamma \, \delta \tag{29}$$

Now we apply assumption 5, and bound

$$|Q^*(s,o^*) \; - \; Q^*(s,o^{b*})| \tag{30}$$

$$\leq \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^k \, \epsilon \, \big(\mathbb{I}(s'_\phi \in G) + \gamma \, W(\phi(s), s'_\phi) \big) +$$

$$\sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^{k+1} \, 2 \, \delta \tag{31}$$

$$= \sum_{k=0}^{\infty} \gamma^k \, \epsilon \, p(s'_\phi \in G) + \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^{k+1} \, \epsilon \, W(\phi(s), s'_\phi) +$$

$$\sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^{k+1} \, 2 \, \delta \tag{32}$$

Since by definition, W is a δ approximation of the value of some states in the MDP, and we know that, in a goal MDP, the maximum value is 1, we can upper bound $W(\phi(s), s'_\phi) \leq (1 + \delta)$, for all $s \in \mathcal{S}, s'_\phi \in \mathcal{S}_\phi$. Resuming,

$$|Q^*(s,o^*) \; - \; Q^*(s,o^{b*})| \tag{33}$$

$$\leq \sum_{k=0}^{\infty} \gamma^k \, \epsilon \; + \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \sum_{k=0}^{\infty} \gamma^{k+1} (\epsilon + \epsilon \, \delta + 2 \, \delta) \tag{34}$$

$$= \frac{\epsilon}{1 - \gamma} + \sum_{s'_\phi \in \mathcal{S}_\phi \setminus \{\phi(s)\}} \frac{\gamma(\epsilon + \epsilon \, \delta + 2 \, \delta)}{1 - \gamma} \tag{35}$$

$$\leq \frac{\epsilon + \gamma \, |\mathcal{S}_\phi| \, (\epsilon + \epsilon \, \delta + 2 \, \delta))}{1 - \gamma} \tag{36}$$

This is a loss bound on executing a single option from the set constituted only by ρ^{b*}. To this option set the results

from (Abel et al. 2020) apply, and in particular equation (3). So we obtain the final result:

$$L(\mathcal{M}, \mathcal{M}_\phi) \leq \frac{\epsilon + \gamma \left|\mathcal{S}_\phi\right| (\epsilon + \epsilon\,\delta + 2\,\delta))}{(1-\gamma)^2} \qquad (37)$$

Other training details

Figure 3b Environment $\mathcal{M}_1$ is the 8-rooms grid appearing in Figure 2. Initial coordinate is $(2, 3)$ (vertical coordinate increases downward). Transitions have 5% failure probability. In the abstract model, actions have 10% failure probability.

- DelayedQ algorithm, with 0.98 discounting, $\epsilon_1 = 0.005$, $\delta = 0.1$, MaxR $= 1.0$, $m = 30$, 500000 timesteps, 10 rollouts per run, 10 runs.
- Q-learning, with 0.98 discounting, learning rate decay from 0.02 to 0.002, 500000 timesteps, ϵ-greedy exploration decay from 1.0 to 0.0.
- Q-learning with our Reward Shaping, with 0.98 discounting, learning rate decay from 0.02 to 0.002, 500000 timesteps, ϵ-greedy exploration decay from 0.5 to 0.0 (Q-learning without reward shaping starting from 0.5 exploration did not converge instead).

Figure 3a Environment $\mathcal{M}_1$ is the 4-rooms grid also appearing in (Abel et al. 2020). Initial coordinate is $(1, 9)$ (vertical coordinate increases downward). Transitions have 5% failure probability. In the abstract model, actions have 10% failure probability.

- DelayedQ algorithm (Strehl et al. 2006), with 0.95 discounting, $\epsilon_1 = 0.01$, $\delta = 0.1$, MaxR $= 1.0$, $m = 30$, 50000 timesteps, 10 rollouts per run, 10 runs.
- Q-learning, with 0.95 discounting, learning rate decay from 0.1 to 0.01, 50000 timesteps, ϵ-greedy exploration decay from 1.0 to 0.0.
- Q-learning with our Reward Shaping, with 0.95 discounting, learning rate decay from 0.1 to 0.01, 50000 timesteps, ϵ-greedy exploration decay from 0.5 to 0.0 (Q-learning without reward shaping starting from 0.5 exploration did not converge instead).

Figure 4 Same 8-rooms environment as in one previous experiment.

- Q-learning with our Reward Shaping, with 0.98 discounting, learning rate decay from 0.02 to 0.002, 500000 timesteps, ϵ-greedy exploration decay from 0.9 to 0.1
- Q-learning, with 0.98 discounting, learning rate decay from 0.02 to 0.002, 500000 timesteps, ϵ-greedy exploration decay from 0.9 to 0.1.

Figure 7 All training algorithms are configured with the parameters. Only the potential differ, depending on the abstraction, as described in main text.

- Q-learning with our Reward Shaping, with 0.98 discounting, learning rate decay from 0.02 to 0.002, 500000 timesteps, ϵ-greedy exploration decay from 0.9 to 0.1

Figure 6 Dueling DQN (Wang et al. 2016), implemented in the TensorForce library. The Q-networks are composed of three dense layers with ReLU activations, which receive input observations as $[x, y, \dot{v}, \cos\theta, \sin\theta]$. The current step for the task represented in Figure 5, which is an automaton state, is passed to the network final layer. Output values are partitioned for each state, and the final layer operates a selection based on its value.