

A Simple Approach for State-Action Abstraction using a Learned MDP Homomorphism

Augustine N. Mavor-Parker
Centre for Artificial Intelligence
University College London
United Kingdom

Andrea Banino
Deepmind
London
United Kingdom

Lewis D. Griffin*
Department of Computer Science
University College London
United Kingdom

Caswell Barry*
Department of Cell and Developmental Biology
University College London
United Kingdom

Abstract

Animals are able to rapidly infer from limited experience when sets of state action pairs have equivalent reward and transition dynamics. On the other hand, modern reinforcement learning systems must painstakingly learn through trial and error that sets of state action pairs are value equivalent—requiring an often prohibitively large amount of samples from their environment. MDP homomorphisms have been proposed that reduce the observed MDP of an environment to an abstract MDP, which can enable more sample efficient policy learning. Consequently, impressive improvements in sample efficiency have been achieved when a suitable MDP homomorphism can be constructed *a priori*—usually by exploiting a practitioner’s knowledge of environment symmetries. We propose a novel approach to constructing a homomorphism in discrete action spaces, which uses a partial model of environment dynamics to infer which state action pairs lead to the same state—reducing the size of the state-action space by a factor equal to the cardinality of the action space. We call this method equivalent effect abstraction. In a gridworld setting, we demonstrate empirically that equivalent effect abstraction can improve sample efficiency in a model-free setting and planning efficiency for model-based approaches. Furthermore, we show on cartpole that our approach outperforms an existing method for learning homomorphisms, while using $33\times$ less training data.

Keywords: Abstraction, Homomorphism, Model-Based, Sample Efficiency

Acknowledgements

Augustine N. Mavor-Parker is supported by the EPSRC project reference 2250955. Caswell Barry was funded by a Wellcome Senior Research Fellowship (212281/Z/18/Z). Augustine N. Mavor-Parker would like to thank Matthew Sargent and Will de Cothi for fruitful discussions on MDP homomorphisms.

*Equal Contribution. Send correspondence to augustine.mavor-parker@cs.ucl.ac.uk

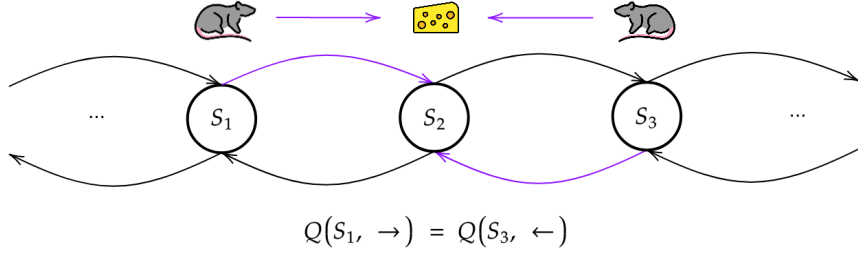


Figure 1: State action pairs that lead to the same state have equivalent values (shown by the blue arrows). Instead of learning these equivalent values individually, we instead learn the value of one abstract action that represents both blue arrows. These values are looked up and learned with a backwards dynamics model during training—meaning no prior knowledge is required from a practitioner.

1 Introduction

Modern reinforcement learning (RL) agents outperform humans in previously impregnable benchmarks such as Go [Silver et al., 2016] and Starcraft [Vinyals et al., 2019]. However, the computational expense of RL hinders its deployment in promising real world applications. In environments with large state spaces, RL agents demand hundreds of millions of samples (or even hundreds of billions) to learn a policy—either within a environment model [Hafner et al., 2020] or by direct interaction [Mnih et al., 2013]. Function approximation can enable some generalisation within a large state space but still most RL agents struggle to extrapolate value judgements to equivalent states.

In contrast, animals can easily abstract away details about states that do not effect their values. For example, a foraging mouse understands that approaching a goal state (shown as a piece of cheese in Figure 1) while travelling east will have the same value as approaching the same goal state from the west. These sort of state abstractions have been defined in RL as Markov decision process (MDP) homomorphisms [Ravindran and Barto, 2001, van der Pol et al., 2020b]. MDP homomorphisms reduce large state action spaces to smaller abstract state action spaces by collapsing equivalent state action pairs in an observed MDP onto a single abstract state action pair in an abstract MDP [van der Pol et al., 2020b].

Given a mapping between an abstract MDP and an experienced MDP, policies can be learned efficiently in the smaller abstract space and then mapped back to the experienced MDP when interacting with the environment [Ravindran and Barto, 2001]. Additionally, if a homomorphic function that maps between equivalent state action pairs is known, it can be used to augment the training experience with more samples. However, obtaining mappings to and from the abstract state action space is challenging. Success has been achieved by hard coding homomorphisms into policy networks but learning homomorphic mappings online is an unsolved problem [van der Pol et al., 2020b].

We develop equivalent effect abstraction, a method to construct MDP homomorphisms given a learned backwards dynamics model—leveraging the fact that state action pairs leading to the same next state must be equivalent. Our approach improves upon the usual model-based-RL scenario as the size of the state action space is fundamentally reduced—enabling more efficient planning (see Figure 3b). Additionally, unlike model-based RL, equivalent abstraction does not require reward modelling. To validate our approach, in Section 4 we demonstrate in a gridworld and control environment that equivalent effect abstraction can improve planning efficiency and sample efficiency.

2 MDP Homomorphisms

Using the definition from [Silver, 2015], an MDP $\mathcal{M}$ can be described by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where $\mathcal{S}$ is the set of all states, $\mathcal{A}$ is the set of all actions, $\mathcal{R} = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$ is the reward function that determines the scalar reward received at each state, $\mathcal{P} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$ is the transition function of the environment describing the probability of moving from state to another for a given action and $\gamma \in [0, 1]$ is the discount factor describing how much an agent should favour immediate rewards over those in future states. An agent interacts with an environment through its policy $\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$ [Silver, 2015] which maps the current state to a given action. To solve an MDP, a reinforcement learning agent must develop a policy that maximises the return G , which is equal to the sum of discounted future rewards $G = \sum_{t=0}^T \gamma^t R_{t+1}$ (where t is the current timestep and T is the number of timesteps in a learning episode). [Ravindran and Barto, 2001] introduced the concept of a homomorphism, $h = \langle \sigma, \{\alpha_s | s \in \mathcal{S}\} \rangle$ from an agent’s experienced MDP $\mathcal{M}$ to an abstract MDP $\bar{\mathcal{M}} = \langle \bar{\mathcal{S}}, \bar{\mathcal{A}}, \bar{\mathcal{P}}, \bar{\mathcal{R}}, \gamma \rangle$, which satisfies two definitions presented by [van der Pol et al., 2020b] in the following form.

$$\bar{\mathcal{P}}(\sigma(s') | \sigma(s), \alpha_s(a)) = \sum_{s'' \in \sigma^{-1}(s')} \mathcal{P}(s'' | s, a), \quad \forall s, s' \in \mathcal{S}, a \in \mathcal{A} \quad (1)$$

$$\bar{\mathcal{R}}(\sigma(s), \alpha_s(a)) = \mathcal{R}(s, a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A} \quad (2)$$

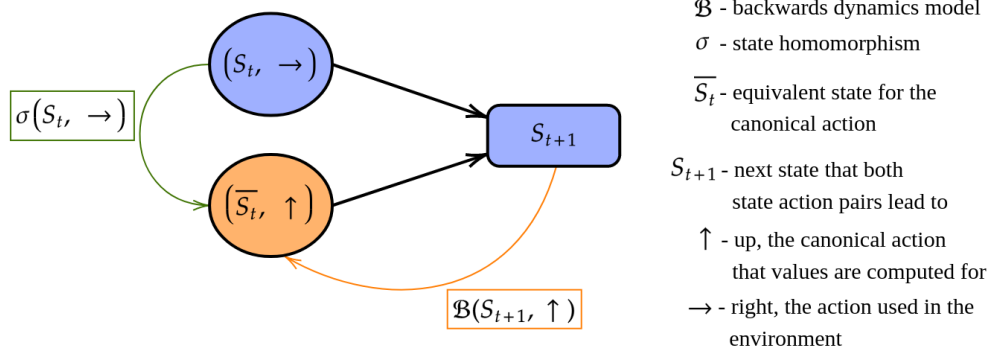


Figure 2: Schematic diagram of how the homomorphic mapping σ is learned. First a backwards model $\mathcal{B}$ is learned from experience mapping every S_{t+1} to its equivalent state for the arbitrarily selected canonical action $\uparrow$ (unless the canonical action was selected in S_t). Once the backwards model $\mathcal{B}$ is learned, we use the states it predicts as labels for a homomorphic map σ , which takes a state action pair as input to predicts the equivalent state for the canonical action $\uparrow$.

Where σ is a mapping from experienced states to abstract states, while α_s is a state dependent mapping between experienced and abstract actions. Equations (1) and (2) demonstrate that a homomorphic map of states and actions must maintain the transitions dynamics and reward functions of the experienced MDP within the constructed abstract MDP. As proved in [Ravindran and Barto, 2001] and leveraged more recently by [van der Pol et al., 2020b], if the true Q-values in the abstract MDP are known $\bar{Q}^*(\sigma(s), \alpha_s(a))$, then equipped with a homomorphism h , the true Q-values in the experienced MDP can be “lifted” [Ravindran and Barto, 2001] from their abstract counterparts.

$$\bar{Q}^*(\sigma(s), \alpha_s(a)) = Q^*(s, a) \quad \forall s \in \mathcal{S}, a \in \mathcal{A} \quad (3)$$

Where Q is the value of observing state s and selecting action a and $*$ signifies the value function is optimal [van der Pol et al., 2020b]. This is useful because the size of abstract state action space $\bar{\mathcal{S}} \times \bar{\mathcal{A}}$ is usually much smaller than the size of the experienced state action space $\mathcal{S} \times \mathcal{A}$. Consequently, if an agent is equipped with a homomorphic map, a policy can be learned efficiently in the reduced abstract state action space and then “lifted” to the experienced state action space when interacting with the environment [Ravindran and Barto, 2001]. Unfortunately, without knowing an environment’s symmetries, obtaining a homomorphic map is difficult. In the next section we present a novel method for inferring a homomorphic map from experience.

3 Equivalent Effect Abstraction

Equivalent effect abstraction is based on a few simple observations that are guaranteed to be true if $\mathcal{R}$ and $\mathcal{P}$ are deterministic. Firstly, state action pairs that lead to the same next state have equivalent reward functions (by definition). Secondly, state action pairs that lead to the same next state also have equivalent transition functions under an unknown state homomorphism function σ . To leverage these properties to learn policies in an abstract MDP, we must develop a method **to learn the mapping σ between state action pairs that lead to the same state**.

Consider the normal RL scenario where an agent takes an action a in state represented by a vector s and ends up in state s' . Once in state s' , how can we infer what other state action pairs could also lead to state s' ? We could make this inference if we had a *backwards* model of the environment. To build a backwards model $\mathcal{B}$, we collect experience tuples $\langle s, a, r', s' \rangle$ from the environment and then optimise the parameters of the model θ with an MSE loss function.

$$\argmin_{\theta} \|\mathcal{B}_{\theta}(s', a) - s\|_2^2 \quad (4)$$

With a learned backwards model $\mathcal{B}_{\theta}$ we can predict what states are equivalent by considering a state s' , sampling actions from our action space and then querying our model to produce a set of equivalent states $\mathcal{S}_{\text{equiv}}^{s'}$ for a given s' .

$$\mathcal{S}_{\text{equiv}}^{s'} = \{\mathcal{B}_{\theta}(s', a) | a \in \mathcal{A}\} \quad (5)$$

Now that we have sets of equivalent state action pairs and we can learn another model σ parametrised by ϕ to map between equivalent states that we have imagined using our backwards model. In practice, we arbitrarily select a canonical action $\bar{a}$ from the full set of experienced actions $\mathcal{A}$. Then we can predict a mapping of all state action pairs to the equivalent state *canonical* action pair using labels generated by $\mathcal{B}_{\theta}$.

$$\argmin_{\phi} \|\sigma_{\phi}(s, a) - \mathcal{B}_{\theta}(s', \bar{a})\|_2^2 \quad (6)$$

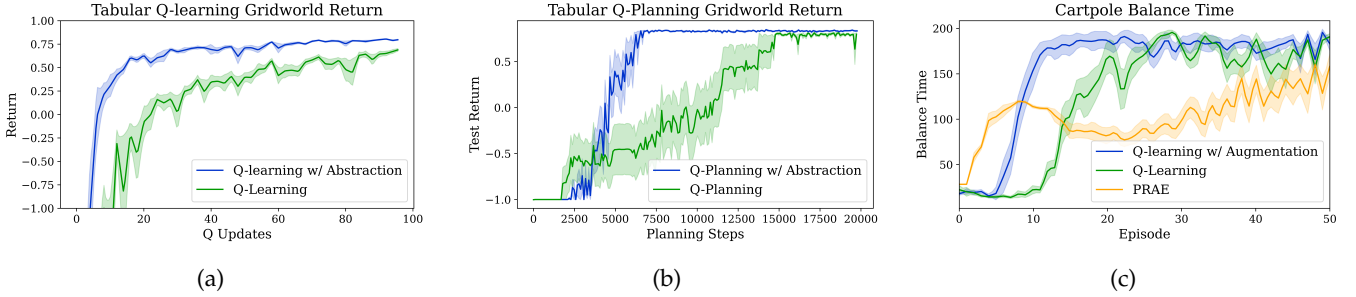


Figure 3: (a), (b) In a tabular gridworld abstracting away redundancies in the Q-table improves sample efficiency. For the vanilla Q-learning/planning baselines, an agent must learn the value of each state action pair for all actions in the action space. With equivalent effect abstraction, we reduce the number of Q-values that need to be learned. This yields improvements in both model-free methods (Q-learning) and model-based approaches (Q-planning). 50 seeds are used for Q-learning while 10 seeds are used for Q-planning. (c) The homomorphic map can be used to generate more samples to improve sample efficiency in cartpole. This approach is the simplest to implement and here delivers plug and play success with a DQN. PRAE [van der Pol et al., 2020a] a related approach for learning homomorphisms is used as baseline. The number random policy steps to collect experience to train the homomorphism are not included here for PRAE or our method. 10 seeds are used for each cartpole method. In all figures, the shaded region represents the standard error of the mean.

In Learning the model of σ_ϕ we can generate the equivalent state canonical action pair for all of the state action pairs in our experienced MDP $\mathcal{M}$. In doing so, we create a new reduced MDP $\tilde{\mathcal{M}}$ where the size of the action space is only one canonical action $\bar{a}$ and hence the total size of the state action space in which we need to find a policy has been reduced by a factor equal to the size of the original action space $\mathcal{A}$. As there is only one action in the reduced MDP, learning the action mapping α_s is trivial. Note that in some cases—such as near borders in gridworlds—there is no equivalent state for a given state-action pair. In these cases we revert to vanilla Q-learning temporarily (see Algorithm 1).

Equipped with our action dependent mapping between equivalent states and actions, which also satisfies Equations (1) and (2), we can now perform more sample efficient reinforcement learning. Note that for simplicity, one could simply use the equivalent states generated by σ_ϕ as data augmentation while training a policy within the experienced MDP $\mathcal{M}$. This is the approach we take in the cartpole experiments in Figure 3c.

Lastly, we highlight that with full knowledge of the transition dynamics of the environment (i.e. a backwards model and a forwards model $\mathcal{F}$) then learning the homomorphic mapping σ_θ is not necessary. The equivalent canonical state action pair for a given state action pair can be obtained by stepping forwards and then backwards through an environment model $\bar{s} = \mathcal{B}(\mathcal{F}(s, a), \bar{a})$. In future work this property could be leveraged for recent RL methods that are bidirectional [Yu et al., 2021]. Finally, for clarity we show exactly how our learned homomorphic map can be integrated into Q-learning to improve sample efficiency in Algorithm 1—an altered version of Q-learning presentations from [Silver et al., 2016][Sutton and Barto, 2018, p. 131].

4 Experiments

We begin with the maze environment from [Sutton and Barto, 2018, p. 202]. We use an open-source Q-learning [Watkins and Dayan, 1992] implementation as a baseline¹ (along with their maze code). We then adapt the Q-learning implementation to make use of a homomorphic map as shown in Algorithm 1. In this particular experiment, we assume a model of the environment is known beforehand—which we move through forwards and then backwards to create a homomorphic map. We show in Figure 3a that our approach compares favorably to vanilla Q-learning, converging much faster to the optimal policy. To demonstrate the potential for improved planning efficiency, we performed a further experiment on the gridworld with model-based Q-planning [Sutton and Barto, 2018, p. 161]. Our Q-planning method is equivalent to the Q-learning implementation but altered to learn by randomly sampling model transitions. At each Q-planning update we pause training and evaluate performance in a 100 step episode. Planning performance is shown in Figure 3b.

Next, we apply our approach to the cartpole benchmark using featurised inputs represented by the four-dimensional vector (position, velocity, angle, angular velocity) and a discrete action space of moving the cart left or right. Prior to learning a policy, we collect 3 episodes of experience with a random policy (data collection episodes are not plotted for PRAE or our method in Figure 3c). With these 3 episodes we are able to train the homomorphic map (two simple linear models, one for each action optimised with Adam [Kingma and Ba, 2015]) to generate equivalent state action pairs

¹<https://github.com/thehawkgriffith/dyna-maze>

for the opposite action to the one experienced. We use this augmentation approach for ease of implementation and an apples to apples comparison to the DQN baseline² (i.e. the policy architectures are equivalent). In the future, equivalent effect abstraction could be used to improve planning efficiency on cartpole when using model-based methods (due to a reduction in the size of the state action space) for even better sample efficiency.

To measure sample efficiency improvements, we trained a DQN augmented with additional experience from our homomorphic map. We also trained the same DQN as a baseline but without the augmented experience. We also compare to an alternative method for learning homomorphisms PRAE [van der Pol et al., 2020a], which uses 100 episodes to learn its homomorphism—33× the amount we use. As shown in Figure 3c, we improve upon both baselines—with our approach converging at around episode 10 while vanilla Q-learning takes around 20 episodes to converge.

5 Related Work

Many approaches aim to learn abstracted models of environment dynamics and a full review of these methods is beyond the scope of an extended abstract. Here we focus on a selected subset of papers that we have found to be particularly relevant to our work. [Ravindran and Barto, 2001] developed a theoretical framework for state action abstraction, while also deriving theoretical guarantees when lifting abstract policies to the experienced MDPs. More recently, homomorphisms in MDPs have been promoted as a means to improve sample efficiency in tasks with inherent symmetries. PRAE [van der Pol et al., 2020a] trained a contrastive world model and accompanying loss function that satisfies the transition dynamics definition (Equation (1)) by design—with some good results in cartpole when 1000 episode of experience are available for training. However, in the relatively low sample regime (e.g. 100 episodes) their homomorphism is less effective as we show in Figure 3c. [van der Pol et al., 2020b] derived a method for learning network weights that are equivariant to environment symmetries (which implicitly creates an MDP homomorphism) but their algorithm requires a practitioner to hardcode such symmetries beforehand. Perhaps the most relevant related approach is the use of afterstates [Sutton and Barto, 2018][p. 136]. Afterstates slightly shift an MDP out of phase with conventional state transitions—creating environment states that are in between the initial effect of a policy’s action and the reaction of the environment to said action. However, applications of the afterstates framework has generally been constrained to boardgames [Tesauro et al., 1995] and usually focus on dealing with the stochasticity of an opponent rather than improved sample efficiency [Antonoglou et al., 2021].

More broadly, model-based RL has enabled superhuman performance in Atari with a relatively small amount of experience [Hafner et al., 2020] but the number of planning steps required to learn a policy is still very large. Backwards models have recently been proposed to improve sample efficiency of world model representation learning, which could be interesting if integrated with equivalent effect abstraction [Yu et al., 2021]. Our approach is related to approaches that learn data augmentation policies for reinforcement learning, however such approaches typically perform best with a set of prespecified data augmentations [Raileanu et al., 2021].

6 Conclusion

Equivalent effect abstraction is a simple method that reduces the size of the state action space considerably for discrete action space MDPs. The approach is easy to implement and provided a backwards dynamics model of the environment it requires no prior knowledge of environment symmetries from a practitioner. In the future, equivalent effect abstraction could be integrated into popular model-based algorithms [Hafner et al., 2020] to improve planning efficiency on large scale tasks or built into state of the art model-free approaches to improve sample efficiency.

Algorithm 1 Equivalent Effect Abstraction Q-learning

```

Set learning rate  $\alpha \in (0, 1]$ , small  $\epsilon$ 
Select a canonical action  $\bar{a}$  arbitrarily from  $\mathcal{A}$ 
Randomly initialise  $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}$ 
Obtain state homomorphism  $\sigma$ 
function GREEDYPOLICY( $s, a$ )
  Initialise  $values = []$ 
  for  $a = 1, |\mathcal{A}|$  do
    if  $\bar{s} = \sigma(s_t, a) \in \mathcal{S}$  then
       $values.append(Q(\sigma(s, a), \bar{a}))$ 
    else
       $values.append(Q(s, a))$ 
    end if
  end for
  return  $a = \operatorname{argmax}_a(values)$ 
end function
for episode = 1,  $M$  do
  Initialise  $s_1$ 
  for  $t = 1, T$  do
    With probability  $\epsilon$  pick random action  $a_t$ 
    otherwise compute  $a_t = \text{GreedyPolicy}(s_t)$ 
    Take action  $a_t$ , observe  $r_t, s_{t+1}$ 
    Set  $a^* = \text{GreedyPolicy}(s_{t+1})$ 
    Set  $Q(s_t, a_t) += \alpha[r_t + \gamma Q(s_{t+1}, a^*) - Q(s_t, a_t)]$ 
    Set  $s_t = s_{t+1}$ 
  end for
end for

```

²<https://gist.github.com/Pocuston/13f1a7786648e1e2ff95bfad02a51521>

References

- [Antonoglou et al., 2021] Antonoglou, I., Schrittwieser, J., Ozair, S., Hubert, T. K., and Silver, D. (2021). Planning in stochastic environments with a learned model. In *International Conference on Learning Representations*. 4
- [Hafner et al., 2020] Hafner, D., Lillicrap, T. P., Norouzi, M., and Ba, J. (2020). Mastering atari with discrete world models. In *International Conference on Learning Representations*. 1, 4
- [Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *ICLR (Poster)*. 3
- [Mnih et al., 2013] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*. 1
- [Raileanu et al., 2021] Raileanu, R., Goldstein, M., Yarats, D., Kostrikov, I., and Fergus, R. (2021). Automatic data augmentation for generalization in reinforcement learning. *Advances in Neural Information Processing Systems*, 34. 4
- [Ravindran and Barto, 2001] Ravindran, B. and Barto, A. G. (2001). Symmetries and model minimization in markov decision processes. 1, 2, 4
- [Silver, 2015] Silver, D. (2015). Lecture notes in reinforcement learning. *Computer Science Department, University College London*. 1
- [Silver et al., 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489. 1, 3
- [Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press. 3, 4
- [Tesauro et al., 1995] Tesauro, G. et al. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68. 4
- [van der Pol et al., 2020a] van der Pol, E., Kipf, T., Oliehoek, F. A., and Welling, M. (2020a). Plannable approximations to mdp homomorphisms: Equivariance under actions. In *AAMAS*. 3, 4
- [van der Pol et al., 2020b] van der Pol, E., Worrall, D., van Hoof, H., Oliehoek, F., and Welling, M. (2020b). Mdp homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210. 1, 2, 4
- [Vinyals et al., 2019] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354. 1
- [Watkins and Dayan, 1992] Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3):279–292. 3
- [Yu et al., 2021] Yu, C., Li, D., Hao, J., Wang, J., and Burgess, N. (2021). Learning state representations via retracing in reinforcement learning. *arXiv preprint arXiv:2111.12600*. 3, 4