

Homomorphisms in Reinforcement Learning Across Continuity and Partial Observability

Arun Tejasvi Chaganty¹

Department of Computer Science and Engineering,
IIT Madras, Chennai, India - 600036

Abstract. Markov Decision Process (MDP) homomorphisms map states and actions from one MDP to another, providing a basis for the transfer of learning, and MDP minimisation. Homomorphisms in the discrete MDP setting have been comprehensively studied; only recently has work been done to define homomorphisms in continuous or partially observable domains. The objective of this work is to address the open questions of homomorphism discovery and MDP minimisation in continuous and/or partially observable MDPs through a unified approach.

1 Introduction

An intuitive approach to tackle complexity in real world problems is to abstract the world with a simpler model that we can handle, and use that model to discover solutions. To do so, we need to understand the conditions for a faithful abstraction, and how to “pull back” solutions from the abstracted model to our real world situation. Homomorphisms in reinforcement learning address precisely these requirements.

Homomorphisms relate the state action spaces of two domains, canonically represented using Markov Decision Processes (MDPs), in a manner that preserves the value function of the agent. Given an MDP M and its homomorphic image M' , there are chiefly four questions we would like to answer:

1. How do we “pull back” solutions from M' to M ?
2. How do we find the homomorphism between M and M' ?
3. What is the smallest MDP isomorphic to M ?
4. Can we find an “approximate” homomorphism between M and M' ?

All four problems have been addressed in the context of discrete MDPs. However, in continuous and partially observable domains, the focus has mainly been on the first question. The objective of this work is to address the remaining questions, hopefully through a unified approach.

We cover the basics of reinforcement learning, in particular describing partially observable MDPs and learning methods in continuous domains in [Section 2](#). The state of the art in homomorphisms is summarised in [Section 3](#), and in particular in [Table 1](#). We outline some approaches for unifying and solving the problems in [Section 4](#). Finally, we describe the domains we wish to evaluate our techniques on in [Section 5](#) and present a timeline for our work in [Section 6](#).

2 Background

In reinforcement learning, the standard representation of an environment and task instance is a Markov decision process (MDP). An MDP can be represented as the tuple, $\langle S, A, P, R, \gamma \rangle$, where S and A are the state and action domains which are known to the agent. $P : S \times A \rightarrow \mathcal{M}(S)$, where $\mathcal{M}(S)$ is the set of all probability measures over S , describes the dynamics of the world through state-action transition probabilities. $R : S \times A \rightarrow \mathbb{R}$ describes the task at hand by ascribing rewards for state transitions. Both P and R are initially unknown to the agent. Finally, $\gamma \in [0, 1]$ is a parameter, called the ‘discount factor’, that weighs the value of future rewards.

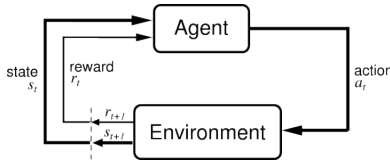


Fig. 1. Agent-Environment Interface

In this setting, an agent in a state $s \in S$ chooses an action $a \in A$, and moves to a state s' with probability $P(s'|s, a)$, receiving a reward $R(s, a)$ (Figure 1). In the fully observable setting, the agent is aware of which state it is in, and the objective of the agent is to find a policy $\pi : S \times \mathcal{M}(A)$, i.e. a decision procedure for selecting actions, that maximises the reward it accumulates in the long run, $R = \sum_i \gamma^i r_i$. R is also called the return. We will discuss the partially observable scenario later in this section.

In the remainder of this section, we will describe the basic approach for finding an optimal policy π for a discrete MDP, and its continuous and partially observable variants.

2.1 Discrete Markov Decision Processes

In the discrete case, S and A are predictably finite sets of states and actions. We define the value function $V^\pi : S \rightarrow \mathbb{R} = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s]$ to be the expected return from s , and $Q^\pi : S \times A \rightarrow \mathbb{R} = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s, A_0 = a]$ to be the expected return from s , after taking the action a . These can be written in a recursive form,

$$V^\pi(s) = \max_a R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V^\pi(s')$$

$$Q^\pi(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) Q^\pi(s', a').$$

Our objective can then be stated as finding a policy π with an optimal value function, i.e. $V^*(s) = \sup_{\pi} V^{\pi}(s)$ at all $s \in S$. The optimal value functions must satisfy the Bellman optimality conditions,

$$V^*(s) = \max_a R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V^*(s')$$

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) \max_{a'} Q^*(s', a').$$

Given an optimal Q , it is possible to construct a greedy policy that is optimal; $\pi(s, a^*) = 1$ when $a^* = \operatorname{argmax}_a Q(s, a)$, and 0 otherwise. In principle, if the agent knew the MDP, it could construct the optimal value function, and from it an optimal policy. However, in the typical setting, the agent is only aware of the state-action space, S and A , and must learn Q through exploration. The Q-learning algorithm learns Q with a simple update for every step the agent takes,

$$Q(s, a) = Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right],$$

where $\alpha \in [0, 1]$ is a parameter that controls the learning rate. It has been shown that the Q-learning algorithm converges to the optimal value function in the limit with fairly permissive assumptions.

2.2 Continuous Markov Decision Processes

In continuous domains, defining S and A must be done with some care. To begin with, S must be a measurable Euclidean space¹; let us denote the Lebesgue measure² by λ . Consider the following (mild) regularity assumption proposed by Antos, Munas and Szepesvari [AMS08],

Property 1. (MDP Regularity) S is a compact subset of the d_S -dimensional Euclidean space, A is a compact subset of $[-A_{\infty}, A_{\infty}]^{d_A}$. The random immediate rewards are bounded by \hat{R}_{\max} and R is uniformly bounded by $R_{\max} : \|r\|_{\infty} \leq R_{\max}$.

We define the evaluation operator; $T^{\pi} : B(S \times A) \rightarrow B(S \times A)$,

$$T^{\pi}Q(s, a) = R(s, a) + \gamma \int_{S, A} ds' da' P(ds'|s, a) \pi(a'|s') Q(s', a').$$

We now define the analogue of the Q -value function recurrence relation, $Q^{\pi} = T^{\pi}Q^{\pi}$. The fixed point of the Bellman operator $T : B(S \times A) \rightarrow B(S \times A)$,

$$TQ(s, a) = R(s, a) + \gamma \int_S ds' \sup_{a' \in A} P(ds'|s, a) Q(s', a'),$$

¹ Roughly, a space X is said to be measurable if a monotonic function $\mu : 2^X \rightarrow \mathbb{R}$ that is additive over finite union can be defined.

² The Lebesgue measure is a generalisation of length/area. It is possible to construct a Lebesgue measure for any Euclidean space.

$Q^* = TQ^*$ is then the optimal value function. By the regularity conditions imposed in [Property 1](#), V^π and Q^π are both bounded by $\frac{R_{\max}}{1-\gamma}$.

In order to approach learning an optimal policy, we must estimate the current value function, Q . A popular scheme is the Fitted Q-iteration approach, wherein the Q function is estimated by regressing on a finite trajectory collected using a stationary policy π_b . Let $[S_t, A_t, R_t]_{1 \leq t \leq N}$, be the dataset, and then Q_{k+1} can be got by,

$$Q_{k+1} = \text{Regress} \left(\left\{ \left[(S_t, A_t), R_t + \gamma \max_{a' \in A} Q_k(X_{t+1}, a') \right]_{1 \leq t \leq N-1} \right\} \right).$$

The regression itself can be solved using a variety of methods, including neural networks and SVMs. A further discussion on suitable regression methods, and a proof of convergence subject to niceness assumptions can be found in [\[AMS08\]](#).

2.3 Partially Observable Markov Decision Processes

The other dimension we wish to explore in this work is partial observability. At the beginning of this section, we described the objective of the agent to be finding an optimal policy to decide which action to select given a state. In the partial observability scenario, the agent does not know which state it is in with absolute certainty, and can only observe some features of the state.

A POMDP [\[KLC98\]](#) is specified by the tuple, $\langle S, \Omega, A, P, O, b, \gamma \rangle$ where Ω is the set of possible observations. $O : A \times S \rightarrow M(\Omega) = O(o|a, s')$ describes the probability of observing $o \in \Omega$ after performing action $a \in A$ and entering state $s' \in S$. As the agent does not know which state it is in, $b \in M(S)$ is a prior belief on this quantity. The other symbols are as defined earlier.

The typical approach to solve POMDPs is to convert the POMDP to a *belief MDP*, $\langle S', A', P', R', \gamma \rangle$, where S' are the belief states of the POMDP. P' and R' can be appropriately defined,

$$P'(b'|b, a) = \mathbb{E}_{b'} \left[\sum_{o \in \Omega} O(o|s', a) \mathbb{E}_b [P(s'|s, a)] \right]$$

$$R'(b, a) = \mathbb{E}_b [R(s, a)].$$

Solving POMDPs is still notoriously difficult. An exact method to solve this MDP assumes the policy to be represented as a non-stationary t-step ‘policy tree’; under this condition, it can be shown that the value function is piecewise linear and convex. A survey of other approaches to solve POMDPs can be found in [\[Mur00\]](#).

Another approach to represent partially observable sequential decision processes is predictive state representations (PSRs) [\[JSR04\]](#). Though PSRs are a very relevant representation for partial observability, discussion on the topic is postponed till a later version.

3 Homomorphisms in RL

A homomorphism is defined in general to be a structure-preserving map. In the context of MDPs, we want a correspondence between the two systems' dynamics (P) and tasks (R). We define a homomorphism g between two MDPs M and \bar{M} to act on their state-action spaces, i.e. $g : S \times A \rightarrow \bar{S} \times \bar{A}$ such that if $g(s, a) = (\bar{s}, \bar{a})$ and $g(s, a') = (\bar{s}', \bar{a}')$, then $\bar{s} = \bar{s}'$. Let $f : S \rightarrow \bar{S}$ be a restriction of g that gives us just the state map, i.e. $f(s) = \bar{s}$ iff $\exists a, a' : g(s, a) = (\bar{s}, \bar{a})$. g is a valid homomorphism iff for every s, a , it satisfies the following two conditions,

$$\begin{aligned} \bar{P}(\bar{s}, \bar{a}, \bar{s}') &= \sum_{s' \in f^{-1}(\bar{s}')} P(s, a, s') \\ \bar{R}(\bar{s}, \bar{a}) &= R(s, a). \end{aligned}$$

There can be other, less strict definitions of homomorphisms based on preserving the value function, policy behaviour, etc. A survey of these definitions can be found in [LWL06].

There are perhaps four serious questions that we wish to answer when talking about homomorphisms,

1. (Transfer) Given a homomorphisms g between MDPs M and M' , can one use a policy learnt in M in M' , and vice versa?
2. (Discovery) Given MDPs M and M' , are they homomorphic?
3. (Minimisation) Given an MDP M , what is the smallest M' isomorphic to M ?
4. (Approximate Minimisation) Given an MDP M , and a class of homomorphisms H , what is the closest approximation $M' \in H(M)$ of M ?

All of these questions have been convincingly answered in the case of discrete MDPs. The objective of this work is to extend these results to continuous or partially observable MDPs, which, thus far have only answers to the transfer problem.

3.1 Transferring Policies in Discrete MDPs

The primary application of MDP homomorphisms is in the transfer of learning in problem to another. Consider an MDP M whose homomorphic image under g is M' . Given a policy π' for M' , the corresponding 'lifted' policy in M is given by,

$$\pi(s, a) = \frac{\pi'(f(s), g(s, a))}{|\{(s, a') : g(s, a') = g(s, a)\}|}.$$

With a slight abuse of notation, we will use $g^{-1}(\pi')$ to refer to this lifted policy.

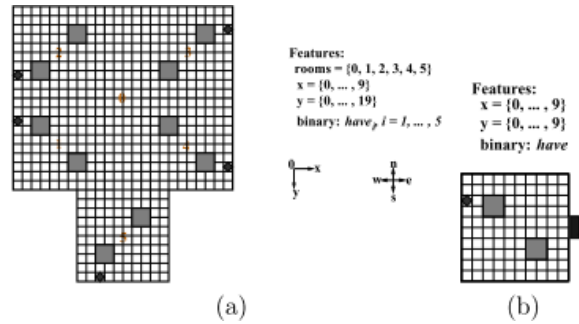


Fig. 2. Relativised Options

Note 1. (Partial Homomorphisms and Relativised Options)

The ability to ‘lift’ policies from one MDP to another can be exploited to solve subproblems in a reinforcement learning domain. Consider the grid-world shown in Figure 2(a) taken from [RB03]; in this domain, there are several rotated copies of the room shown in Figure 2(b). For each room in (a), let us define a homomorphism mapping each cell within the room to those of the template room (b). All states outside the room are mapped to the ‘sink’ state, denoted by the black square in (b). We can now define *options* [SPS99] for each room, using the optimal policy for (b), π , and the above homomorphisms; $\mathcal{O} = \{g_r^{-1}(\pi) \mid \text{for each room } r\}$.

3.2 MDP Minimisation and Homomorphism Discovery in Discrete MDPs

In order to answer the minimisation question, Narayanamurthy and Ravindran reduced the MDP homomorphism query to a graph automorphism problem in [NR08] by constructing an equivalent weighted digraph from an MDP $M\langle S, A, P, R \rangle$.

We briefly describe the reduction: Construct a graph G_M , with S as nodes. For each node s , add an edge to the node s' if there is some action a that takes the agent from s to s' . Each edge is weighted with the vectors, $\langle p_{a_1}, \dots, p_{a_{|A|}} \rangle$ and $\langle r_{a_1}, \dots, r_{a_{|A|}} \rangle$, where $p_{a_i} = P(s, a_i, s')$, and $r_{a_i} = R(s, a_i, s')$. We could also view this as a construction of two graphs, one for P , and the other for R ; the graph isomorphisms we are looking for belong to the common subset.

There are cases for which using the symmetry group may not suffice to capture the symmetries present. **TODO: Describe such an example.**

3.3 Approximate Homomorphisms for Discrete MDPs

An exact symmetry may not exist in some domains, for example **TODO: figure here**, however, excluding certain states would allow for symmetry. Another reason to consider approximate homomorphisms is the computational complexity of finding exact homomorphisms.

State aggregation (using partition from homomorphism); Block transition probabilities. Can bound loss dependent on maximum difference between the two images. (Shravan)

Another is weighted graph matching, min. error (using max-norm) (Dattani)

Note 2. (Soft Homomorphisms) Relaxation to a QP problem. Mapping between discrete and continuous. Stretch homomorphisms. Does not handle SDAR

3.4 Homomorphisms for Continuous MDPs

Definition - Satinder Singh - Limitations

3.5 Homomorphisms for POMDPs

Definition - Pippin - Limitations - Reductions to PSRs?

3.6 Summary

	Definition	Transfer	Discovery	Minimisation	Approximate	Minimisation
Discrete						
Continuous						
Partially Obs.						

Table 1. Summary of Homomorphisms in RL: Open Problems

The discussion of this section can be summarised in [Table 1](#).

4 Approaches

4.1 Reduction of POMDPs to continuous MDPs

4.2 Automorphism Group

Generators

4.3 Embedding in a Real Algebraic Variety

What about factored MDPs?

4.4 Generating Equivalent MDPs from Samples

4.5 Spectral Methods for Approximations

5 Evaluation

I propose to use the following domains for evaluation:

- **Cartpole:** Continuous state space, for exact homomorphisms.
- **Acrobot:** Continuous state space, for exact homomorphisms.
- **Pinball:** Continuous state space, for approximate homomorphisms.
- **Bouncing Ball:** Discrete state space, with noise (partially observable), for exact homomorphisms.
- **Bouncing Ball:** Continuous state space, with noise (partially observable), for exact homomorphisms.

6 Timeline

- **Dec. 8th - Dec. 26th:** Implement domains, and existing algorithms for them. Empirically experiment with symmetries. Work on mathematical formalism.
- **Dec. 26th - Jan. 9th:** Mathematically refine algorithms, and sketch their implementations.
- **Jan. 9th - Jan. 30th:** Implement algorithms, tune, and revise.
- **Jan. 30th - Feb. 13th:** Evaluate comprehensively.
- **Feb. 13th - Feb. 24th:** Write up.
- **Feb. 24th:** ICML submission.

References

- AMS08. Antos Andra, Remi Munos, and Csaba Szepesvári. Fitted Q-iteration in continuous action-space MDPs. In *NIPS*, pages 1–8, 2008.
- JSR04. Michael R James, Satinder Singh, and Matthew R Rudary. Predictive State Representations : A New Theory for Modeling Dynamical Systems. In *UAI*, pages 512–519, 2004.
- KLC98. Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, May 1998.
- LWL06. Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a Unified Theory of State Abstraction for MDPs. In *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics*, pages 531—539, 2006.
- Mur00. Kevin P Murphy. A Survey of POMDP Solution Techniques. Technical report, 2000.
- NR08. Shравan Matthur Narayanamurthy and Balaraman Ravindran. On the hardness of finding symmetries in Markov decision processes. *Proceedings of the 25th international conference on Machine learning - ICML '08*, (1):688–695, 2008.

- RB03. Balaraman Ravindran and Andrew G Barto. Relativized Options : Choosing the Right Transformation. In *International Conference on Machine Learning*, 2003.
- SPS99. Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and Semi-MDPs : Learning , Planning , and Representing Knowledge at Multiple Temporal Scales at Multiple Temporal Scales. *Artificial Intelligence*, 112:181–211, 1999.