Towards a Unified Theory of State Abstraction for MDPs

Lihong Li Thomas J. Walsh Michael L. Littman

Department of Computer Science
Rutgers University,
Piscataway, NJ 08854
{lihong,thomaswa,mlittman}@cs.rutgers.edu

Abstract

State abstraction (or state aggregation) has been extensively studied in the fields of artificial intelligence and operations research. Instead of working in the *ground* state space, the decision maker usually finds solutions in the *abstract* state space much faster by treating groups of states as a unit by ignoring irrelevant state information. A number of abstractions have been proposed and studied in the reinforcement-learning and planning literatures, and positive and negative results are known. We provide a unified treatment of state abstraction for Markov decision processes. We study five particular abstraction schemes, some of which have been proposed in the past in different forms, and analyze their usability for planning and learning.

1 Introduction

State abstraction (or state aggregation) has been widely studied in artificial intelligence (e.g., [10]) and operations research [25] as a technique for accelerating decision making. Abstraction can be thought of as a process that maps the *ground* representation, the original description of a problem, to an *abstract* representation, a much more compact and easier one to work with [10]. In other words, abstraction allows the decision maker to distinguish *relevant* information from *irrelevant* information. From a computational perspective, state abstraction is a technique for making learning and planning algorithms practical in large, real-world problems.

In this paper, we will focus on state abstraction in Markov decision processes [22], where different types of abstraction have been proposed, including bisimulation [11], homomorphism [23], utile distinction [18], and policy irrelevance [14]. Given the diversity of choices, natural questions arise such as:

- Is there a unified definition of state abstraction?
- What are the relationships among these abstractions?
- How does a solution to the abstract MDP relate to the ground MDP? What guarantees can be made?
- How can we select among abstraction schemes?

With the different types of abstractions being developed rather independently, a unified treatment is not yet available. Although there are results for estimating and bounding approximation errors with state abstraction [1, 2, 30], we seek to answer qualitative questions like "What information is lost when an abstraction is applied?" and "When is the optimal policy still preserved?" In fact, lacking such qualitative insights about state abstractions can lead to negative results when using state abstraction. For example, McCallum [18] has observed that aggregating states using one form of state abstraction makes it impossible to find the optimal policy using value iteration [22] or Q-learning [31]; Gordon [12] reported a chattering phenomenon of SARSA(λ) [26] when combined with improperly constrained state abstraction.

The major contribution of this paper is a general treatment of state abstraction that *unifies* many previous works on the topic. Instead of providing immediately available algorithms for solving more concrete problems, we will focus on the abstraction theory itself, including formal definitions, relationships, and qualitative properties (such as preservation of optimality and learnability).

The rest of the paper is organized as follows. Section 2 presents notation and reviews previous work.

Section 3 studies state abstraction formally and examines five particular types of abstraction in some detail. Section 4 discusses concrete examples and illustrates how these abstractions affect the size of the abstract state space. Finally, we give some concluding remarks and mention future research directions in Section 5.

2 Background & Prior Work

In this paper, we will focus on sequential decision making problems and the concrete model of Markov decision processes (MDPs) [22]. An MDP can be described as a five-tuple $\langle S, A, P, R, \gamma \rangle$ where S is a finite set of states; A is a finite set of actions; P is the transition function with $P^a_{ss'}$ denoting the next-state distribution after taking action a in state s; R is a bounded reward function with R^a_s denoting the expected immediate reward gained by taking action a in state s; and $\gamma \in [0, 1]$ is a discount factor.

A policy is a mapping from states to actions: $\mathcal{S} \mapsto \mathcal{A}$. Given a policy π, we define the state-value function, $V^\pi(s)$, as the expected cumulative reward received by executing π from state s. Similarly, the state-action value function, $Q^\pi(s, a)$, is the expected cumulative reward received by taking action a in state s and following π thereafter. A reinforcement-learning agent [27] attempts to *learn* an optimal policy π^* whose value functions are denoted by $V^*(s)$ and $Q^*(s, a)$, respectively. It is well-known that $V^* = \max_\pi V^\pi$ and $Q^* = \max_\pi Q^\pi$.

Given the full model of an MDP (i.e., the five-tuple), a set of standard algorithms exists for finding the optimal value function as well as the optimal policy, including linear programming, value iteration, and policy iteration [22]. However, if the transition and/or reward functions are unknown, the decision maker has to gather information by interacting with the environment. In fact, a reinforcement-learning agent is able to compute the optimal policy by such interaction [27].

2.1 Previous Work

The problem of defining state abstraction rules has been the focus of previous work. Boutlier et al. [3] introduced an algorithm, stochastic dynamic programming, that built aggregation trees in the factored setting to create an abstract model where states with the same transition and reward functions under a fixed policy were grouped together. This work was identified by Givan et al. [11] as a special case of their own form of abstraction, bisimulation, where states with the same transition and reward functions are aggregated. Given a flat representation of an MDP, this partitioning can be accomplished in polynomial time.

Although such partitioning provides a compact representation of the ground MDP, bisimulation is a very strict criterion for aggregation. As such, several adaptations have been proposed. Ravindran et al. [23] examined state aggregation based on homomorphisms of the model, rather than strict action matching. They also propounded the use of options [29] as a mechanism for encapsulating abstraction information. We note here that such a vehicle can be used independently of the type of abstraction being performed. Givan et al. [5] analyzed approximate bisimulation, which drops the exact equivalence requirement in favor of a bound, as a method for producing boundedly accurate MDPs (BMDPs). Similarly, Ferns et al. [8] proposed bisimulation metrics, which use statistical semi-metrics [9] to determine the similarity between two states' transition functions. This measure is then combined with the difference in the reward functions to decide aggregation. We note that this criterion comes very close to comparing the actual value functions of the two states, an idea we will return to shortly.

Several of these algorithms can be used to aggregate states between iterations of model-based planning algorithms like value iteration or policy iteration. This approach is reminiscent of a previously proposed adaptive aggregation strategy [2], which groups states in between runs of value iteration based on Bellman residuals. This algorithm was unique both in its criterion and its allowance for states to be aggregated or disaggregated at every abstraction step. Dietterich's MAXQ hierarchy [6] effectively aggregates states within a subtask only if their reward and transition functions are the same for any policy consistent with the hierarchy. McCallum [18] proposed a less strict criterion, aggregating only states that have the same optimal action and similar Q-values for these actions, based on a statistical test. McCallum allowed such aggregation to occur while the system was learning the MDP itself. Using statistics to do the abstraction online was a feature of earlier works including the G-algorithm [4], which aggregated states with the same reward and Q-values for each action.

A more recent online aggregation strategy using statistical tests was provided by Jong and Stone [14], which groups states based on "Policy

Abstraction Mechanism	**Criterion**	**Exactness**	**MDP given?***	**Notes**
Bisimulation [11]	Model equivalence	Exact	Yes	Strictest measure
Homomorphisms [23]	Model equivalence	Exact, matching of actions flexible	Yes	Accounts for spacial relations (e.g. symmetry)
Approximate Bisimulation [5]	Model similarity	Bounded	Yes	Builds BMDPs
Bisimulation Metrics [8]	Model similarity	Statistically tested	Yes	Error bounds deducible
MAXQ [6]	Model equivalence for hierarchically consistent policies	Exact	Yes	Integrated into the MAXQ hierarchy
Stochastic Dynamic Programming [3]	Equivalent models given a policy	Exact	Yes	Covered by bisimulation
G Algorithm [4]	Equivalent rewards and Q-values	Statistically tested	No	Each feature's relevance must be independent
Utile Distinction [18]	Equivalent best actions with similar Q-values	Statistically tested	No	Aggregation occurs online
Policy Irrelevance [14]	Equivalent best actions	Statistically tested	No	May not yield optimal policy for ground MDP
Adaptive Aggregation [2]	Similar Bellman residuals	Bounded	Yes	States can be (dis)aggregated dynamically

* I.e., does abstraction discovery require the MDP model?

Table 1: Selected previous strategies for state aggregation

Irrelevance"—states are aggregated if they have the same optimal action. However, if such abstract MDPs are used to learn a policy for the larger MDP, they may not yield optimal policies [14, 18] and may even prevent some algorithms from converging [12].

A summary of the properties of the aforementioned work is presented in Table 1. The table orders the algorithms roughly from strictest to coarsest abstractions. It has been noted that coarser abstractions methods, although providing a better potential for computational efficiency and generalization, have looser performance loss bounds on the value function [30]. These methods overlap considerably, and in this paper we endeavor to formalize a unified theory of abstraction to better understand these similarities, and the differences.

3 A State Abstraction Theory

We adopt the viewpoint of Giunchiglia and Walsh [10], who argue that abstraction is in general *a mapping from one problem representation to a new representation, while preserving some properties.* In this work, we focus on the preservation of properties that are needed for an agent to make decisions that lead to optimal behavior. Previous abstraction definitions have applied this insight. For example, bisimulation is essentially a type of abstraction that preserves the one-step model of an MDP (i.e., the transition and reward functions), while policy-irrelevance types of abstraction attempt to preserve the optimal actions. Many of the methods we have cited attempt "fuzzy conservation" of such properties through statistical tests and notions of bounding, but in the interest of developing a formalism we choose to focus on abstraction schemes where states are only aggregated when they have exact equality over the parameters of the abstraction scheme.

3.1 A General Definition of State Abstraction

The definition of state abstraction in MDPs we give here is not novel. However, based on this definition, we are able to study formal properties of abstraction itself in subsequent sections.

Definition 1 *Let* $M = \langle S, A, P, R, \gamma \rangle$ *be the* ground *MDP and its* abstract *version be* $\bar{M} = \langle \bar{S}, A, \bar{P}, \bar{R}, \gamma \rangle$. *Define the* abstraction function *as* $\phi : S \mapsto \bar{S}$; $\phi(s) \in \bar{S}$ *is the abstract state corresponding to ground state* s, *and the* inverse image $\phi^{-1}(\bar{s})$, *with* $\bar{s} \in \bar{S}$, *is the set of ground states that correspond to* $\bar{s}$ *under abstraction function* ϕ. *Note that under these assumptions,* $\{\phi^{-1}(\bar{s}) \mid \bar{s} \in \bar{S}\}$ *partitions the ground state space* S. *To guarantee* $\bar{P}$ *and* $\bar{R}$ *are well-defined, a weighting function is needed:* $w : S \mapsto [0, 1]$, *where for each* $\bar{s} \in \bar{S}$, $\sum_{s \in \phi^{-1}(\bar{s})} w(s) = 1$. *With these definitions at hand, we can define the transition and reward* *functions of the abstract MDP as follows:*

$$\begin{aligned}\bar{R}^a_{\bar{s}} &= \sum_{s\in\phi^{-1}(\bar{s})} w(s)R^a_s,\\ \bar{P}^a_{\bar{s}\bar{s}'} &= \sum_{s\in\phi^{-1}(\bar{s})}\sum_{s'\in\phi^{-1}(\bar{s})} w(s)P^a_{ss'}.\end{aligned}$$

It is easy to verify that $\bar{P}^a_{\bar{s}\bar{s}'}$ is a well-defined next-state distribution by checking that $\sum_{\bar{s}'}\bar{P}^a_{\bar{s}\bar{s}'}$ sum up to 1 for any a and $\bar{s}\in\bar{S}$:

$$\begin{aligned}\sum_{\bar{s}'}\bar{P}^a_{\bar{s}\bar{s}'} &= \sum_{\bar{s}'}\sum_{s\in\phi^{-1}(\bar{s})}\sum_{s'\in\phi^{-1}(\bar{s})} w(s)P^a_{ss'}\\ &= \sum_{s\in\phi^{-1}(\bar{s})} w(s)\sum_{s'\in S}P^a_{ss'}\\ &= \sum_{s\in\phi^{-1}(\bar{s})} w(s) \;=\; 1.\end{aligned}$$

Intuitively, $w(s)$ measures the extent to which a state s contributes to the abstract state $\phi(s)$. In the rest of the paper, we will only mention w when necessary; otherwise, it can be any valid weighting function.

Next, we consider how policies $\bar{\pi}$ in the abstract MDP translate to policies π in the ground MDP. Since all ground states in $\phi^{-1}(s)$ are treated identically, it is natural to translate policies by the following rule: $\pi(s,a)=\bar{\pi}(\phi(s),a)$ for all s and a.

Finally, value functions for the abstract MDP $\bar{M}$ can be defined in the straightforward way and are denoted $V^{\bar{\pi}}(\bar{s})$, $V^*(\bar{s})$, $Q^{\bar{\pi}}(\bar{s},a)$, and $Q^*(\bar{s},a)$, respectively.

3.2 Topology of Abstraction Space

In this subsection, we are concerned with the question of how these abstractions relate to each other. Let Φ_M denote the abstraction space—the set of abstractions on MDP M. We will need the following definition.

Definition 2 *Suppose $\phi_1,\phi_2\in\Phi_M$. We say ϕ_1 is* finer *than ϕ_2, denoted $\phi_1\succeq\phi_2$, iff for any states $s_1,s_2\in S$, $\phi_1(s_1)=\phi_1(s_2)$ implies $\phi_2(s_1)=\phi_2(s_2)$. If, in addition, $\phi_1\neq\phi_2$, then ϕ_1 is* strictly finer *than ϕ_2, denoted $\phi_1\succ\phi_2$. We may also say ϕ_2 is* (strictly) coarser *than ϕ_1, denoted $\phi_2\preceq\phi_1$ ($\phi_2\prec\phi_1$). We say ϕ_1 and ϕ_2 are* comparable *if either $\phi_1\succeq\phi_2$ or $\phi_2\succeq\phi_1$.*

It is obvious from Definitions 1 and 2 that the relation $\succeq$ satisfies reflexibility, antisymmetry, and transitivity. Therefore, we arrive at the following theorem.

Theorem 1 *The finer relation $\succeq$ is a partial ordering.*

In other words, if we depict the topology of the abstraction space Φ_M as a graph, we would obtain a directed acyclic graph (DAG) with a self-loop at each node; each edge points from an abstraction ϕ_1 to another abstraction $\phi_2\preceq\phi_1$. At the very end is the finest representation (the ground representation) we denoted ϕ_0 (i.e., ϕ_0 is the identity mapping and $\bar{S}=S$), and at the other extreme is the coarsest representation (the null representation consisting of a single abstract state).

3.3 Five Types of Abstraction

There are many abstractions for an MDP, because there are many possible ways to partition the state space. However, not all abstractions appear to be equally important. A useful abstraction has to preserve some information that is critical for solving the original MDP. In this subsection, we will see how five pieces of important information in MDPs lead to respective abstractions, as defined below. [1]

Definition 3 *Given an MDP $M=\langle S,A,P,R,\gamma\rangle$, and any states $s_1,s_2\in S$, we define five types of abstraction as below, with an arbitrary but fixed weighting function $w(s)$.*[2]

1. *A* model-irrelevance abstraction ϕ_{model} *is such that for any action a and any abstract state $\bar{s}$, $\phi_{\text{model}}(s_1)=\phi_{\text{model}}(s_2)$ implies $R^a_{s_1}=R^a_{s_2}$ and $\sum_{s'\in\phi^{-1}_{\text{model}}(\bar{s})}P^a_{s_1s'}=\sum_{s'\in\phi^{-1}_{\text{model}}(\bar{s})}P^a_{s_2s'}$.*
2. *A* Q^π-irrelevance abstraction ϕ_{Q^π} *is such that for any policy π and any action a, $\phi_{Q^\pi}(s_1)=\phi_{Q^\pi}(s_2)$ implies $Q^\pi(s_1,a)=Q^\pi(s_2,a)$.*
3. *A* Q^*-irrelevance abstraction ϕ_{Q^*} *is such that for any action a, $\phi_{Q^*}(s_1)=\phi_{Q^*}(s_2)$ implies $Q^*(s_1,a)=Q^*(s_2,a)$.*
4. *An* a^*-irrelevance abstraction ϕ_{a^*} *is such that every abstract class has an action a^* that is optimal for all the states in that class, and $\phi_{a^*}(s_1)=\phi_{a^*}(s_2)$ implies that $Q^*(s_1,a^*)=\max_a Q^*(s_1,a)=\max_a Q^*(s_2,a)=Q^*(s_2,a^*)$.*

[1] Similar definitions are possible with state value functions, but we focus on state-action value functions because of their key role in popular planning and learning algorithms.

[2] Although w does not appear important in the definition, it can play an important role in affecting planning and/or learning efficiency [30].

5. *A* π^*-irrelevance abstraction ϕ_{π^*} *is such that every abstract class has an action* a^* *that is optimal for all the states in that class, that is* $\phi_{\pi^*}(s_1) = \phi_{\pi^*}(s_2)$ *implies that* $Q^*(s_1, a^*) = \max_a Q^*(s_1, a)$ *and* $Q^*(s_2, a^*) = \max_a Q^*(s_2, a)$.

Intuitively, ϕ_{model} preserves the one-step model (e.g., bisimulation [11]); ϕ_{Q^π} preserves the state-action value function for all policies; ϕ_{Q^*} preserves the optimal state-action value function (e.g., stochastic dynamic programming with factored representations [3] or the G-algorithm [4]); ϕ_{a^*} preserves the optimal action and its value (e.g., utile distinction [18]); and ϕ_{π^*} attempts to preserve the optimal action [14]. [3]

3.4 Properties of the Abstractions

In the previous subsection, we introduced a collection of state abstractions. Below we study their properties formally, which provides insights into some previous findings.

The first question we address is how the abstractions relate to one another. The following theorem states that they form a chain under the partial order $\succeq$. Furthermore, there exist examples showing that they are not equal to each other in general.

Theorem 2 *For any MDP, we have* $\phi_0 \succeq \phi_{\text{model}} \succeq \phi_{Q^\pi} \succeq \phi_{Q^*} \succeq \phi_{a^*} \succeq \phi_{\pi^*}$.

As a consequence, any one of the five abstractions is a special case of other finer abstractions. For example, an instance of ϕ_{Q^*} is also an instance of ϕ_{a^*}- and ϕ_{π^*}-irrelevance abstraction, but is not necessarily a ϕ_{model}- or ϕ_{Q^π}-irrelevance abstraction. This observation is helpful when we consider the learning and planning problems below, that is, if we prove some property of an abstraction, this property automatically applies to the finer abstractions.

The second question is related to planning. More specifically, given an abstract MDP $\bar{M}$, we may use standard dynamic-programming algorithms such as value iteration and policy iteration to solve it, obtaining an optimal abstract policy $\bar{\pi}^*$. With what abstractions will $\bar{\pi}^*$ be guaranteed to be optimal in the ground MDP M? We have the following results:

Theorem 3 *With abstractions* ϕ_{model}, ϕ_{Q^π}, ϕ_{Q^*}, *and* ϕ_{a^*}, *the optimal abstract policy* $\bar{\pi}^*$ *is optimal in the ground MDP. However, there exists examples where the optimal policy with abstraction* ϕ_{π^*} *is suboptimal in the ground MDP [14] (see also Figure 1(b)).*

Proof (sketch) We are able to show that the value-iteration operator on the state-action value functions in the abstract MDP is a contraction mapping, of which the optimal value function is the fixed point [22]. The optimality then follows from the fact that the optimal actions as well as their Q-values are preserved after abstraction. □

Finally, we consider the learning problem, where the agent estimates the optimal value function based on experience. In the abstraction case, Q-learning requires some modification. After observing a transition (s_t, a_t, r_t, s_{t+1}), the agent does the following backup in the abstract state-action space:

$$Q(\phi(s_t), a_t) \quad \xleftarrow{\alpha_t} \quad r_t + \gamma \max_{a'} Q(\phi(s_{t+1}), a'),$$

where α_t is an appropriately decaying step-size parameter such that $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$. Using standard tools in stochastic approximation theory [13, 17, 24], we obtain the following results for Q-learning:

Theorem 4 *Assume that each state-action pair is visited infinitely often and the step-size parameters decay appropriately.*

1. *Q-learning with abstractions* ϕ_{model}, ϕ_{Q^π}, *or* ϕ_{Q^*} *converges to the optimal state-action value function in the ground MDP. Therefore, the resulting optimal abstract policy is also optimal in the ground MDP.*

2. *Q-learning with abstraction* ϕ_{a^*} *does not necessarily converge. However, if the behavior policy is* fixed, *Q-learning converges to* $\bar{Q}^*$ *with respect to some weighting* $w(s)$, *and the greedy policy is optimal in the ground MDP, although* $\bar{Q}^*$ *may not predict the optimal values for suboptimal actions in the ground MDP.*

3. *Q-learning with abstraction* ϕ_{π^*} *can converge to an action-value function whose greedy policy is suboptimal in the ground MDP. However, we note that policy-search methods (e.g. [28]) may still find the optimal policy in this case.*

Proof (sketch) The first part of the theorem follows earlier convergence results [17, Theorem 1], since the conditions in the theorem are all satisfied. For the second part, since we assume that each

[3] We note that several of these examples are not the coarsest possible implementations of their respective abstractions, including those listed for ϕ_{Q^*} and ϕ_{π^*}.

(s, a) is visited infinitely often and a fixed behavior policy is being followed, the learner must run into a steady state-occupation distribution $\mu(s) > 0$ for all $s \in S$. The weighting function $w(s)$ associated with ϕ_{a^*} is in proportion to $\mu(s)$ and fixed in the limit. So, the resulting abstract MDP is well-defined and stationary. The convergence of Q-learning then follows from previous analyses under these conditions. If the behavior policy is not fixed, however, we have empirically observed a chattering phenomenon similar to what Gordon observed [12], as illustrated in Figure 1(c). The third part of the theorem comes from existing examples [14] or the one shown in Figure 1(b). More details are given below. □

The negative learning results described above for ϕ_{a^*} and ϕ_{π^*} stem from similar causes, although ϕ_{π^*} introduces a pitfall beyond non-convergence: convergence to a suboptimal policy. We investigate both cases through the examples in Figure 1. First, there is the possibility under ϕ_{a^*} or ϕ_{π^*} that Q-learning will fail to converge. An example MDP and abstraction, along with a sample run of Q-learning under a periodically changing policy is provided in Figures 1(a and c). For ϕ_{a^*} this malady can only afflict the suboptimal actions, but in ϕ_{π^*} all state action pairs are susceptible. Removing action 1 from the aggregate state in Figure 1(a) provides such an example. Gordon [12] observed this behavior for the combination of SARSA and ϕ_{π^*}. One might think that because Q-learning's backups are off-policy, this behavior would not be possible. However, the chattering phenomena is actually caused not by the learning algorithm, but by partial observability introduced by the abstraction function (or more generally, function approximator).

Suppose, using ϕ_{a^*}, we combine two state-action pairs that have different Q-values (say, (s_1, a) and (s_2, a)), making the Q-value of the abstract state: $Q(\bar{s}, a) = w(s_1) \cdot Q(s_1, a) + w(s_2) \cdot Q(s_2, a)$. If these weights are based on state occupation probabilities and a changing policy is used (as in our example) these weights may not come from a stationary distribution, causing Q-learning to chatter. The cause of chatter under ϕ_{π^*} is similar, but the weighted sum above does not hold strictly in this case. Thus, neither ϕ_{a^*} nor ϕ_{π^*} can guarantee Q-learning convergence unless a fixed policy is used. To our knowledge, this is the first time such behavior has been catalogued for Q-learning or for an abstraction scheme guaranteed to preserve the Q-values of the optimal actions. We also note that our results hold not just for Q-learning, but for any learning algorithm based on on-trajectory sampling, combined with a non-stationary policy.

ϕ_{π^*} introduces a new liability for Q-learning beyond the possible chattering of optimal action Q-values. Under ϕ_{π^*} it is possible for Q-learning to converge to values that indicate an optimal action in $\bar{M}$ that is suboptimal in M. This shortcoming of policy irrelevance has been noted previously by others [14, 18]. The crux of this failure is that the combination of two such states can lead to a Q-value for the aggregate state, $Q' = Q^*(\bar{s}, a^*) > Q^*(s, a^*)$, if the state s is aggregated with another state that has a higher Q-value. Since Q' corresponds to the optimal action, it will be used to define the Q-values of other state-action pairs recursively through the Bellman equation. Hence in the one-step case, as in Figure 1(b), since $r_1 = R^{a_1}_{s_0} > r_2 = R^{a_2}_{s_0}$ and hence $r_1 + \gamma Q' > r_2 + \gamma Q'$, a_1 is the optimal action in s_0 of $\bar{M}$, but indeed is suboptimal in M. We note this concern is not limited to the one-step backup case, but to all state-action pairs that receive a backup from $\bar{s}$ (i.e., all states with a path leading to $\bar{s}$). Also notice that suboptimal action selection is impossible under ϕ_{a^*} because under this abstraction scheme, $Q' = Q^*(\bar{s}, a^*) = Q^*(s, a^*)$.

In contrast to model-free reinforcement learning (e.g., Q-learning) is model-based learning (e.g., Prioritized Sweeping [19]), where the agent first builds an empirical model to approximate the MDP through interaction experience, and then solves this model. In the context of state abstraction, the agent first builds an empirical abstract model of $\bar{M}$ and then solves for $\bar{\pi}^*$, which will be translated back into a ground policy. For such a class of learning algorithms, we have the following results:

Theorem 5 *With abstractions ϕ_{model}, ϕ_{Q^π}, ϕ_{Q^*}, and ϕ_{a^*}, the empirical model built from experience converges to the true abstract model with infinite experience, if the weighting function $w(s)$ is fixed. Furthermore, model-based reinforcement learning converges to the optimal abstract value function whose greedy policy is optimal in the ground MDP. For ϕ_{π^*}, optimality is not guaranteed in general.*

Proof (sketch) If the weighting function is fixed, the abstract MDP is well-defined, regardless of the policy used to collect the empirical data, and the empirical model converges asymptotically to the true abstract MDP given enough samples. Therefore, with abstractions ϕ_{model}, ϕ_{Q^π}, ϕ_{Q^*}, and ϕ_{a^*}, the optimal policy in the empirical abstract MDP converges to the optimal policy in the true abstract MDP, which in turn is optimal in the ground MDP

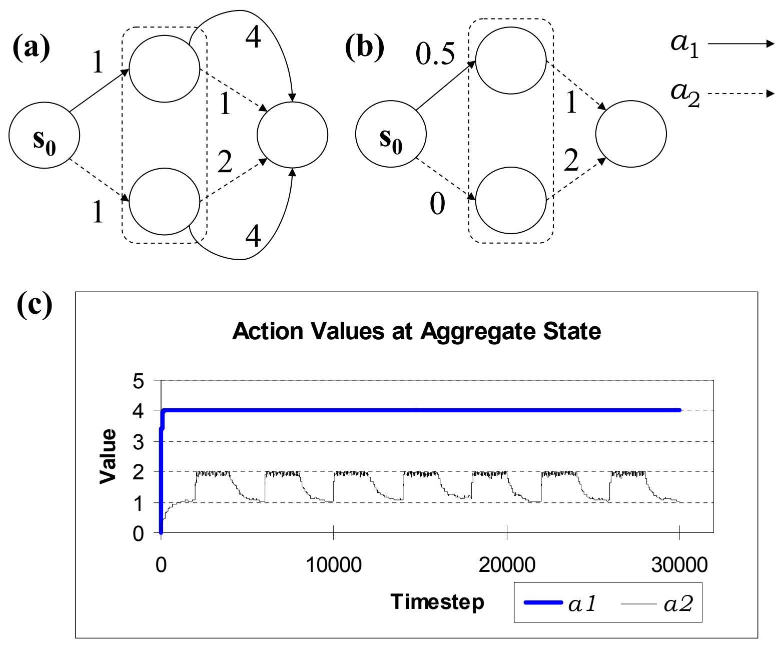

Figure 1: (a) An example problem where ϕ_{a^*} is applied but Q-learning may fail to converge. The solid and dashed lines represent actions 1 and 2 respectively. The corresponding graph shows the value function in the aggregated (middle) state for each of its actions. The policy is changed every 2000 steps from "do action 1 w.p. $1-\epsilon$" to "do action 2 w.p. $1-\epsilon$". With probability $\epsilon = .1$, a random action is selected. Notice the instability (chattering) of the suboptimal action value. (b) An MDP that when subjected to ϕ_{π^*} yields an optimal policy for $\bar{M}$ that is suboptimal in M. Both of these examples are modifications of the MDP investigated by Gordon [12].

by Theorem 3. With abstraction ϕ_{π^*}, the empirical abstract model is inherently flawed (c.f. Figure 1(b)), and therefore, a planning algorithm may fail to find the optimal policy in the ground MDP. □

3.5 Necessary Conditions for Locally Checkable Abstraction

Since ϕ_{a^*} is the coarsest abstraction function listed in Theorem 5 as being sufficient for convergence in a model-based setting for convergence to a policy that is also optimal in the ground MDP, one may speculate whether the conditions defining ϕ_{a*} are also *necessary* for this guarantee. This harkens back to an open question posed by McCallum [18] asking if Utile Distinction (his instantiation of ϕ_{a^*}) is a necessary and sufficient condition for preserving the optimal policy. In a limited sense, we find that the answer is "yes". We define the set of *locally checkable* abstractions, Φ_{lc} to be those that use only the one-step reward and transition, the abstract class of the one-step reachable states, and the optimal Q-function values of the two candidate states to decide whether these two states should be aggregated. Notice that Φ_{lc} contains all the abstraction functions defined in Definition 3. Investigating this class, we come to the following conclusions:

Theorem 6 *Within Φ_{lc}, ϕ_{a^*} is the coarsest possible abstraction with which it is guaranteed that model-based reinforcement learning algorithms under the conditions in Theorem 5, or Q-learning with a fixed policy, will converge to a value function whose greedy policy is optimal in the ground MDP.*

Theorem 6 follows from the fact that if even one of the optimal action's Q-values were changed, we can construct a malicious example in the spirit of Figure 1(b) where the new Q-value causes a state in a trajectory leading to the aggregated state to promote a sub-optimal action in M to optimality in $\bar{M}$. This lower bound on the coarseness stands as necessary in this class because reasoning on the transition or reward function from the states being aggregated cannot ensure optimality without also ensuring the preservation of the optimal action's Q-value.

3.6 Summary

We have introduced a general definition of state abstraction and used it to derive formal properties of several natural types of abstraction. Given a number of choices of abstractions, an important question is: *which one shall we prefer?* As can be seen from the analysis above, more and more information about the problem (the MDP) is lost with coarser and coarser abstractions. For example, ϕ_{model} gives the opportunity to *recover* essentially the entire model, while it is only possible to represent and learn the optimal state-action value function with ϕ_{Q^*}. The coarser ϕ_{a*} ceases to guarantee learnability of the value function for suboptimal actions, but does allow for planning (i.e., value iteration). Finally, at the very end of the chain, even optimal planning is generally lost with ϕ_{π^*}, although an optimal policy is still representable. However, a coarser abstraction results in a larger reduction in the state space, which in turn translates into efficiency of solving the problem. Therefore, there is a tradeoff between *minimizing information loss* and *maximizing state space reduction* when selecting abstractions. In the next section, we will examine several examples that will shed some light on how much state space reduction could be achieved with these abstractions.

4 Case Studies

Here, we consider three concrete, widely studied examples, and study how different types of abstraction affect the size of abstract state space. Since it is an open question how to find Q^π-irrelevance abstractions efficiently without enumerating all possible policies, we will not give results for ϕ_{Q^π} below. However, we should note that the size of the abstract state space with ϕ_{Q^π} is between the sizes of abstract state spaces with ϕ_{model} and ϕ_{Q^*}.

The first problem is the TAXI domain [6], where a taxi is to navigate in a 5×5 grid world, pick up a passenger and then deliver her to the destination. There are 500 states in the ground state space and 6 actions. The agent is charged a reward of -1 for each action and a final $+20$ for successfully delivering the passenger to her destination.

Another problem is known as the COFFEE domain [3], several versions of which have been studied in the literature. We consider a simplified scenario where a robot is wandering around five locations: lab, office, coffee room, mail room, and hallway. Its goal is three-fold: to tidy up the lab, to deliver coffee and mail to the user sitting in the office. There are a total of 400 ground states and the robot is able to perform 7 actions (although not all actions are allowed in all states). It receives a -1 reward upon taking any single action, and is rewarded when coffee and mail are delivered, and when the lab is tidy. We use two reward functions and obtain two versions of this problem.

The last problem is called BITFLIP, variations of which have been studied under other names [11, 16]. In this problem, the agent is given a binary vector of 10 bits. Its goal is to reach the zero vector by flipping bits that are 1's; it is illegal to flip a bit that is 0. At any moment it has three actions to choose from: flip_highest_bit, flip_lowest_bit, and flip_random_bit, each of which results in an immediate reward of -1. If the bit to flip, say bit k, is the highest bit (namely, all bits $j > k$ are 0's), it is reset to 0 and other bits are unchanged; otherwise, all higher bits $j \geq k$ are set to 1 and lower bits $i < k$ remain unchanged. The optimal action in all states is flip_highest_bit and the optimal value of a state is the negative of the number of 1's in the state.

The sizes of the *coarsest* abstract state space of each type of abstraction in all three problems are shown in Table 2. Note that ϕ_0 corresponds to no abstraction, and the coarsest studied abstraction, ϕ_{π^*}, always has a size no greater than $|A|$. The results have several implications. First, ϕ_{model} and ϕ_{Q^*} do not appear to provide great opportunity for state-space reduction, compared to ϕ_{a^*} and ϕ_{π^*}. In the taxi domain, for example, the abstract state space is (almost) as large as the ground state space.

Second, ϕ_{π^*} seems to provide the greatest opportunity of state space reduction. However, as we have shown in previous theorems, ϕ_{π^*} does not guarantee that dynamic programming and Q-learning will converge to the optimal policy and/or value function.

These two observations, when combined together, and also considering Theorem 5, imply that ϕ_{a^*} abstraction may be a useful type of abstraction, which not only possesses guaranteed optimality but also leads to significant state space reduction.

5 Conclusions

We have introduced a theory of state abstraction for Markov decision processes. In particular, we gave a formal definition of state abstraction and

domains	ϕ_0	ϕ_{model}	ϕ_{Q^*}	ϕ_{a^*}	ϕ_{π^*}
TAXI (5×5)	500	500	489	381	6
COFFEE (v1)	400	296	256	124	7
COFFEE (v2)	400	296	256	132	7
BITFLIP (10)	1024	513	257	11	1

Table 2: Sizes of abstract state spaces.

analyzed the properties of general abstractions as well as some specific abstractions. Equipped with this theory, more insights and a better understanding are obtained. We have provided the beginnings of a formal foundation and classification for state abstraction, which we hope will spur more efforts in formal analysis for state abstraction and novel powerful algorithms.

One important problem we did not address is the *discovery* problem. That is, given a model or interaction experience, how does the agent discover a specific abstraction? Efficient discovery algorithms exist for some abstractions such as bisimulation [11]; however, it is not obvious at present how to efficiently discover some other types of abstraction, for example, ϕ_{Q^π}, without exhaustively computing all value functions. Givan et al. [11] voiced such concerns regarding abstraction measures based on Q-functions. However, there has been significant work in approximate tests for value-function-based irrelevance notions [8, 18]. So, although we concede that the discovery problem for these abstractions is an open problem, we believe this avenue of research merits further exploration, particularly given our results regarding ϕ_{Q^*} and ϕ_{a^*}. We would like to emphasize that discovering abstractions can be important for solving sets of related MDPs. For example, abstractions can be a form of knowledge transferred across a set of related tasks [14].

We have only presented a set of basic results of the abstraction theory. In fact, there are several interesting extensions. First, the abstractions in Definition 3 are based on exact equivalence of certain quantities. This definition is often too stringent in practice, especially in stochastic domains and discounted MDPs. One possible extension is to relax the exactness and allow some form of approximate or soft abstraction [5, 7, 8]. Second, we can extend the definition of abstraction using action homomorphisms [23]. Third, it is important to see how state abstraction theory could be used in hierarchical reinforcement learning [6, 21, 29]. Fourth, there could be theoretical interest in investigating operations on abstractions, such as intersection and composition. Finally, we would like to mention a connection between state abstraction and factored representations [3] as well as state-space partitioning in continuous MDPs [15, 20].

Acknowledgements

We gratefully acknowledge the support of the National Science Foundation (IIS-0325281), DARPA IPTO (HR0011-04-1-0050), and NASA. Carlos Diuk provided the solution to the taxi problem. We also thank Nick Jong for helpful discussions on policy irrelevance.

References

[1] James C. Bean, John R. Birge, and Robert L. Smith. Aggregation in dynamic programming. *Operations Research*, 35(2):215–220, 1987.

[2] Dimitri P. Bertsekas and David A. Castañon. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Transactions on Automatic Control*, 34(6):589–598, 1989.

[3] Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1–2):49–107, 2000.

[4] David Chapman and Leslie Pack Kaelbling. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *IJCAI-91*, pages 726–731, 1991.

[5] Thomas Dean, Robert Givan, and Sonia M. Leach. Model reduction techniques for computing approximately optimal solutions for Markov decision processes. In *UAI-97*, pages 124–131, 1997.

[6] Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

[7] Eyal Even-Dar and Yishay Mansour. Approximate equivalence of Markov decision processes. In *COLT-03*, pages 581–594, 2003.

[8] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite Markov decision processes. In *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 162–169, 2004.

[9] Alison L. Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435, 2002.

[10] Fausto Giunchiglia and Toby Walsh. A theory of abstraction. *Artificial Intelligence*, 57(2–3):323–390, 1992.

[11] Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 147(1–2):163–223, 2003.

[12] Geoffrey J. Gordon. Chattering in Sarsa(λ). Technical report, CMU Learning Lab, 1996.

[13] Tommi Jaakkola, Michael I. Jordan, and Satinder P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6(6):1185–1201, 1994.

[14] Nicholas K. Jong and Peter Stone. State abstraction discovery from irrelevant state variables. In *IJCAI-05*, 2005.

[15] Lihong Li and Michael L. Littman. Lazy approximation for solving continuous finite-horizon MDPs. In *AAAI-05*, pages 1175–1180, 2005.

[16] Michael L. Littman, Carlos Diuk, and Alexander L. Strehl. A hierarchical approach to efficient reinforcement learning. In *Proceedings of the ICML-05 Workshop on Rich Representations for Reinforcement Learning*, 2005.

[17] Michael L. Littman and Csaba Szepesvári. A generalized reinforcement-learning model: Convergence and applications. In *ICML-96*, pages 310–318, 1996.

[18] Andrew McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, University of Rochester, Rochester, NY, 1995.

[19] Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103–130, 1993.

[20] Andrew W. Moore and Christopher G. Atkeson. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21(3):199–233, 1995.

[21] Ronald Parr and Stuart J. Russell. Reinforcement learning with hierarchies of machines. In *Advances of Neural Information Processing Systems 10*, pages 1043–1049, 1998.

[22] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, New York, 1994.

[23] Balaraman Ravindran and Andrew G. Barto. SMDP homomorphisms: An algebraic approach to abstraction in semi-Markov decision processes. In *IJCAI-03*, pages 1011–1016, 2003.

[24] Herbert Robbins and Sutton Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[25] David F. Rogers, Robert D. Plante, Richard T. Wong, and James R. Evans. Aggregation and disaggregation techniques and methodology in optimization. *Operations Research*, 39:553–582, 1991.

[26] Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems 8*, pages 1038–1044, 1996.

[27] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, March 1998.

[28] Richard S. Sutton, David McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063, 2000.

[29] Richard S. Sutton, Doina Precup, and Satinder P. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999.

[30] Benjamin van Roy. Performance loss bounds for approximate value iteration with state aggregation. *Mathematics of Operations Research*. To appear.

[31] Christopher J.C.H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, University of Cambridge, UK, 1989.