Transferring State Abstractions Between MDPs

Thomas J. Walsh THOMASWA@CS.RUTGERS.EDU
Lihong Li LIHONG@CS.RUTGERS.EDU
Michael L. Littman MLITTMAN@CS.RUTGERS.EDU
Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854

Abstract

Decision makers that employ state abstraction (or state aggregation) usually find solutions faster by treating groups of states as indistinguishable by ignoring irrelevant state information. Identifying irrelevant information is essential for the field of knowledge transfer where learning takes place in a general setting for multiple environments. We provide a general treatment and algorithm for transferring state abstractions in MDPs.

1. Introduction

The fields of "transfer learning" and state abstraction are closely related. In transfer learning, an agent attempts to reuse knowledge from several source domain instances in one or more (possibly before unseen) target domain instances. This goal requires some notion of a "generalized" or "abstract" state space where reasoning can be done and applied to any instance in the set. Thus, deciding what knowledge to transfer between environments can be construed as determining the correct state *abstraction* scheme for a set of source instances and then applying this compaction to a target instance. In this paper, we present an algorithm for accomplishing this abstraction transfer in structured Markov Decision Processes (MDPs) (Puterman, 1994) based on prior work that formalized abstraction in MDPs (Li et al., 2006) and prior work in transferring information for acting optimally in multiple MDPs (Guestrin et al., 2003; Jong & Stone, 2005).

2. Definitions

MDPs 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; R is a bounded reward function; and $\gamma \in [0, 1]$ is a discount factor. In this paper we deal with structured MDPs, where the state space is comprised of n multi-valued features $f_1, \cdots, f_n$. A policy, $\pi : \mathcal{S} \mapsto \mathcal{A}$, maps states to actions. 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 (Sutton & Barto, 1998) attempts to *learn* an optimal policy π^* with value function $Q^*(s, a)$. An abstraction is a function $\phi : S \mapsto \bar{S}$, where S is the *ground* state space and $\bar{S}$ the *abstract* state space.

We define "transfer learning" in structured MDPs in terms of a distribution D over possible *target* MDPs, which share some common features (such as states defined in terms of the same variables or relations). Using a set $\mathcal{M}_s$ of m *source* MDPs sampled from D, we wish to uncover some information that allows us to maximize the following *speedup ratio*:

$$\frac{\mathbf{E}_{M_t \sim D} \left\{ T(M_t) \right\}}{\mathbf{E}_{\mathcal{M}_s \sim D, M_t \sim D} \left\{ T(M_t | \mathcal{M}_s) \right\}}$$

where $T(M_t)$ is the time needed to find an optimal policy in a target MDP, M_t, and $T(M_t|\mathcal{M}_s)$ is the time needed to find an optimal policy if information is transferred from $\mathcal{M}_s$. In this paper, we focus on learning and transferring MDP abstractions.

3. Prior Work

In this section we review several intuitive definitions of abstraction for MDPs, present several key properties of these abstractions and discuss previous attempts to achieve knowledge transfer via abstractions.

3.1. Five Types of Abstraction

Defining state-abstraction rules for MDPs has been the focus of previous work, including a recent unified treatment of the problem (Li et al., 2006), which provides an exhaustive list of previously proposed abstraction techniques as well as defining five abstrac-

Appearing in the *ICML-06 Workshop on Structural Knowledge Transfer for Machine Learning*, June 29, Pittsburgh, PA, Copyright 2006 by the author(s)/owner(s).

tion schemes based on seemingly important features of MDPs: ϕ_{model}, ϕ_{Q^π}, ϕ_{Q^*}, ϕ_{a^*}, and ϕ_{π^*}. Intuitively, ϕ_{model} preserves the one-step model (e.g., bisimulation (Givan et al., 2003)); 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 (Boutilier et al., 2000), the G-algorithm (Chapman & Kaelbling, 1991), Symbolic Dynamic Programming (Boutilier et al., 2001; Großmann et al., 2002), or Q-RRL (Dzeroski et al., 2001)); ϕ_{a^*} preserves the optimal action and its value (e.g., utile distinction (McCallum, 1995)); and ϕ_{π^*} preserves the optimal action (e.g. Policy Irrelevance (Jong & Stone, 2005) or P-RRL (Dzeroski et al., 2001)). [1]

3.2. Abstraction Properties

Here, we present several known theoretical results for the five abstraction schemes, focusing on how they affect planning and learning algorithms. We refer the reader to previous work (Li et al., 2006) for the proofs as well as an expanded analysis of the abstraction schemes, including empirical evaluations.

First, we consider whether standard dynamic-programming algorithms such as value iteration and policy iteration, when applied to the abstract MDP $\bar{M}$ yield an optimal policy $\bar{\pi}^*$ that maps to an optimal policy in the ground MDP.

Theorem 1 *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, an optimal policy with abstraction ϕ_{π^*} may be suboptimal in the ground MDP.*

Next, we consider the problem of *learning* the value function, where the agent estimates the optimal value function based on experience. We extend the standard Q-learning update to the abstract MDP as:

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

Theorem 2 *Q-learning with abstractions ϕ_{model}, ϕ_{Q^π}, or ϕ_{Q^*} converges and yields a policy that is optimal in the ground MDP. Q-learning with abstraction ϕ_{a^*} does not necessarily converge, but will converge with a fixed behavior policy; in either case, it yields a value function whose greedy policy is optimal in the ground MDP. Q-learning with abstraction ϕ_{π^*} can converge to an state-action value function whose greedy policy is suboptimal in the ground MDP.*

A more detailed discussion of this result, with examples and an extension to model-based reinforcement learning, is provided in previous work (Li et al., 2006). Since these abstractions are defined independently of the language used to represent the state space, these results are applicable to domains specified in any language, including logical ones (Dzeroski et al., 2001; Boutilier et al., 2001). The preceding results are consistent with the negative results for ϕ_{π^*} in the state abstraction (Gordon, 1996; Jong & Stone, 2005) and relational reinforcement learning (RRL) literature (Dzeroski et al., 2001).

3.3. Abstraction Transfer in Prior Work

Many of the positive transfer-learning results have appeared in the RRL literature (Dzeroski et al., 2001; Guestrin et al., 2003), where a state space is described using a relational or even full first-order language. Although the richer representations do facilitate knowledge transfer, the practicality of these measures is incumbent upon the abstraction techniques used to compact these often large state spaces. For example, in the relational extensions of value iteration (Boutilier et al., 2001; Großmann et al., 2002) and Q-learning (Dzeroski et al., 2001), a variant of ϕ_{Q^*} is used to keep the backups from expanding the state space far beyond the size needed to represent the value function.

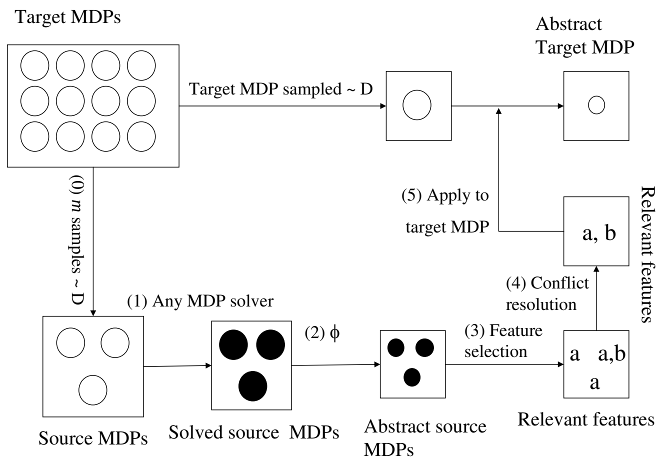
Our algorithm for abstraction transfer is closely related to previous work on discovering irrelevant state variables with policy irrelevance (Jong & Stone, 2005). Our main contribution is extending this work to the complete range of abstractions discussed in Section 3.1 and developing theoretical guarantees for the behavior of traditional planning and learning algorithms in the induced abstract target environments, dependent on the choice of abstraction, ϕ.

4. The GATA Algorithm

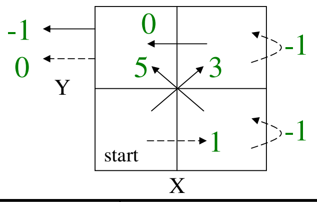
4.1. Motivation and Overview

In RRL, abstract value functions are often transferred to new domain instances, so in essence, both the value function and the abstraction are transferred. But, transferring the source values themselves often just bootstraps the value function in the target instance, where learning or planning algorithms must still be used to find the true optimal policy. Also, previous work has indicated that transferring value functions may not be very general in the real world, as they depend critically on problem sizes (Dzeroski et al., 2001). Instead, we can induce a speedup by just transferring the learned abstraction scheme from the source environments to the target, and then planning or learning in this smaller space. More formally, given a set of m source MDPs as described earlier, we wish to apply an

[1]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^*}.



(a)



Variables	Action outcomes/rewards
X[0-1]	As shown ⟶
Y[0-1]	As shown - - - -›
Color[0…d-1]	ColorUp: (Color + 1) mod d Reward = (-1 * Color) –1
Counter[0…d-1]	CounterUp: (Counter + 1) mod d Reward = –1

(b)

Figure 1. (a) GATA: The General Abstraction Transfer Algorithm; (b) The example world used in Section 5.

abstraction technique to compact each source instance without sacrificing the ability to plan or learn optimally. Then, we wish to unify these source abstractions and apply this unified abstraction to the target environment. The General Abstraction Transfer Algorithm (GATA), illustrated in Figure 1 (a) implements this abstraction transfer.

4.2. Implementation

In Step (1), the source MDPs are solved using a planning or learning algorithm. In Step (2), an abstraction scheme, such as ϕ_{Q^*}, is applied to each of the solved source MDPs to produce abstract source MDPs. We note that if Step (2) involves an abstraction scheme that does not require solving or learning the MDP (e.g. ϕ_{model} given the MDP), then Step (1) is not necessary, but value-function-based abstraction schemes often yield greater compaction (Li et al., 2006). In Step (3), we construct a set of *relevant* features, F, and irrelevant features, $\bar{F}$, for each abstract MDP. To avoid problems due to feature correlation (e.g. aliasing) this construction is performed incrementally. A feature f is considered *irrelevant* with respect to abstraction ϕ if and only if the abstraction that ignores f and the features currently in $\bar{F}$, denoted $\phi_{f\cup\bar{F}}$, is finer than ϕ. That is, $\phi_{f\cup\bar{F}}(s_1) = \phi_{f\cup\bar{F}}(s_2)$ implies $\phi(s_1) = \phi(s_2)$. If so, f is added to $\bar{F}$, otherwise it is relevant and added to F. Notice we are finding the relevant features consistent with ϕ in each MDP, not necessarily the minimum features needed to preserve the ability to learn or plan optimally. In Step (4), we resolve conflicts over relevance between the multiple sources (multiple F's) in a "safe" manner where a feature is deemed "relevant" for the target environment if it was tagged as relevant in *any* of the source environments. In Step (5), the resulting feature-based abstraction function (which ignores all features except those determined to be relevant in Step (4)) is used to compact a target MDP. The applicability of the optimal policy learned in such reduced target MDPs is discussed below.

4.3. Discussion and Theoretical Results

As long as m (the number of source MDPs) is large, the following theorems hold (proofs omitted due to space constraints):

Theorem 3 *Replacing ϕ in GATA with abstractions ϕ_{model}, ϕ_{Q^π}, ϕ_{Q^*}, or ϕ_{a^*}, the optimal policy in the abstract target MDP, $\bar{\pi}^*$, is optimal in the ground target MDP. However, an optimal policy with abstraction ϕ_{π^*} may be suboptimal in the ground target MDP.*

Theorem 4 *Using Q-learning in the abstract target MDP produced by GATA we have the following cases:*

1. *If ϕ is replaced by ϕ_{model}, ϕ_{Q^π}, or ϕ_{Q^*}, Q-learning in the resulting abstract target MDP converges to the optimal state-action value function and policy in the ground target MDP.*
2. *If ϕ is replaced by ϕ_{a^*}, Q-learning in the resulting abstract target MDP does not necessarily converge. However, it converges when using a* fixed *behavior policy. In either case, it yields a value function whose greedy policy is optimal in the ground target MDP.*
3. *If ϕ is replaced by ϕ_{π^*} Q-learning in the resulting abstract target MDP can converge to an state-action value function whose greedy policy is suboptimal in the ground target MDP. However, policy search methods may still be effective.*

These results help justify the admittedly daunting discovery times for ϕ_{Q^*} and ϕ_{a^*} (which involve determining the optimal value functions for the ground source MDPs). Essentially, we are willing to pay the high price for discovery of ϕ_{Q^*} or ϕ_{a^*} if this provides information on what state variables (in the factored case) or relations (in the RRL case) can be ignored for planning or learning optimally in the target MDP.

5. An Example Domain

We demonstrate the power of GATA and the pitfalls promulgated in the previous two theorems using a simple grid world and three of our abstraction schemes (ϕ_{Q^*},ϕ_{a^*},and ϕ_{π^*}). The source instance, made up of four features: X, Y, a "counter", and the color of the lighting, is depicted in Figure 1 (b). All MDPs in this class have four state variables and four actions, but each instance has a different limit, d, on the dimensionality of Color and Counter. The actions CounterUP and ColorUP are available at any location. We denote the feature-based abstraction scheme developed by GATA with parameter ϕ as $A(\phi)$. The *relevant* features uncovered by $A(\phi_{Q^*})$,$A(\phi_{a^*})$,and $A(\phi_{\pi^*})$ are [Color, Y, X],[Y, X] and [X] respectively. Since the targets can have d values for Color and Counter, the state space sizes for the abstractions are: none $O(d^2)$; $A(\phi_{Q^*})$ $O(d)$; $A(\phi_{a^*})$ $O(1)$; $A(\phi_{\pi^*})$ $O(1)$. Although $A(\phi_{\pi^*})$ actually gives the greatest compaction, when this abstraction ("only use X") is applied to a target MDP, standard dynamic programming algorithms used on the abstract space will yield a policy that is suboptimal in the ground targets, forcing our speedup ratio to 0. In contrast, and consistent with Theorem 3, $A(\phi_{Q^*})$ and $A(\phi_{a^*})$ will yield abstraction schemes such that planning in the abstract target space yields a policy optimal for the ground target, and ensures our speedup ratio is no less than 1. More generally, we note that $A(\phi_{a^*})$ provides the smallest abstract models consistent with guaranteed optimal planning. Learning results are similar (Theorem 4).

6. Conclusions & Future Work

We have presented an algorithm for transferring abstractions learned in source MDPs to target MDPs. We also provided results concerning the soundness of planning and learning algorithms in the abstract target MDPs induced by this algorithm.

The theoretical results of this paper rely on having enough samples, m, to make sure all features relevant in any MDP in D are discovered. Studying the effectiveness of GATA based on m and the similarity of the MDPs in D (in terms of how often and to what degree each feature is relevant) is an area for future research. Also, the feature-selection algorithm we outlined (Step (3)) is only guaranteed to produce the *minimal* set of relevant features if all combinations of feature values represent valid states. Restricting the search space to keep such sets near minimal in all cases is an area of future research. Replacing ϕ in GATA with an abstraction scheme that allows for inexact matching (e.g. Bisimulation metrics (Ferns et al., 2004)) could produce an abstraction that results in a bounded loss in the abstract value function for the target MDPs. Similarly, a less cautious combination of relevant features in Step (4) could result in a lossy but still generally effective abstraction scheme.

Finally, we thank the support of the DARPA Transfer Learning program.

References

Boutilier, C., Dearden, R., & Goldszmidt, M. (2000). Stochastic dynamic programming with factored representations. *Artificial Intelligence*, *121*, 49–107.

Boutilier, C., Reiter, R., & Price, B. (2001). Symbolic dynamic programming for first-order MDPs. *IJCAI* (pp. 690–700).

Chapman, D., & Kaelbling, L. P. (1991). Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. *IJCAI* (pp. 726–731).

Dzeroski, S., Raedt, L. D., & Driessens, K. (2001). Relational reinforcement learning. *Machine Learning*, *43*, 7–52.

Ferns, N., Panangaden, P., & Precup, D. (2004). Metrics for finite Markov decision processes. *UAI* (pp. 162–169).

Givan, R., Dean, T., & Greig, M. (2003). Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, *147*, 163–223.

Gordon, G. J. (1996). *Chattering in Sarsa*(λ) (Technical Report). CMU Learning Lab.

Großmann, A., Hölldobler, S., & Skvortsova, O. (2002). Symbolic dynamic programming within the fluent calculus. *IASTED International Conference Artificial and Computational Intelligence* (pp. 378–383).

Guestrin, C., Koller, D., Gearhart, C., & Kanodia, N. (2003). Generalizing plans to new environments in relational MDPs. *IJCAI*.

Jong, N. K., & Stone, P. (2005). State abstraction discovery from irrelevant state variables. *IJCAI*.

Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. *9th International Symposium on Artif. Intel. and Math.*.

McCallum, A. (1995). *Reinforcement learning with selective perception and hidden state*. Doctoral dissertation, University of Rochester, Rochester, NY.

Puterman, M. L. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. New York: Wiley.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.