Representation Discovery in Sequential Decision Making

Sridhar Mahadevan
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
mahadeva@cs.umass.edu

Abstract

Automatically constructing novel representations of tasks from analysis of state spaces is a longstanding fundamental challenge in AI. I review recent progress on this problem for sequential decision making tasks modeled as Markov decision processes. Specifically, I discuss three classes of representation discovery problems: finding functional, state, and temporal abstractions. I describe solution techniques varying along several dimensions: diagonalization or dilation methods using approximate or exact transition models; reward-specific vs reward-invariant methods; global vs. local representation construction methods; multiscale vs. flat discovery methods; and finally, orthogonal vs. redundant representation discovery methods. I conclude by describing a number of open problems for future work.

Introduction

A common practice in designing AI systems in many areas is to assume that human designers provide the essential knowledge structures such as features or a task hierarchy, constraining the search space where optimal or satisficing solutions may be found; the machine implements an efficient search strategy for finding solutions within the given space. This division of labor between human and machine is sensible for single-task environments, where considerable engineering and fine-tuning of the input representation is possible. It becomes increasingly difficult to maintain this paradigm for agents that are faced with solving a novel collection of problems. Representation discovery is an area of AI research that involves designing methods for automatically constructing novel representations of tasks that facilitate their solution. Early research on representation discovery in AI includes the pioneering work of Saul Amarel (1968), who advocated designing agents that discover novel representations through global analysis of state spaces. By finding bottlenecks and symmetries, Amarel outlined ways in which agents could collapse and shrink state spaces. Amarel focused on deterministic state space problems, such as the missionaries and cannibal problem.

In this paper, I focus on representation discovery methods for *stochastic* problems, which are more representative of real-world applications such as robotics and scheduling. In particular, I summarize recent work on autonomous representation discovery in the area of sequential decision making based on Markov decision processes (MDPs) and their variants. MDPs are widely used in *operations research* (Bertsekas and Tsitsiklis 1996; Puterman 1994), *probabilistic planning* (Boutilier, Dean, and Hanks 1999), *reinforcement learning* (Sutton and Barto 1998), and *robot learning* (Connell and Mahadevan 1993).

I describe three categories of representation discovery problems in MDPs: finding functional abstractions, state abstractions, and temporal abstractions. Functional abstractions correspond to finding a compressed representation of the space of functions on a state (action) space, such as reward functions, state transition functions, and value functions. State abstractions partition the state space into disjoint sets that preserve some property, such as respecting rewards and transition dynamics, the optimal policy, or the optimal value function. Temporal abstractions are based on discovering task hierarchies, which enable multiscale approaches to solving MDPs. I review solution methods to these problems along four dimensions: diagonalization vs. dilation methods; reward-sensitive vs. reward-independent methods; flat vs. multiscale methods; and finally, orthogonal vs. redundant representation discovery methods. I conclude with a list of challenges that constitute key directions for further research.

Markov Decision Processes

Markov decision processes (MDPs) are a widely used model of sequential decision making in AI (Puterman 1994). An MDP $M = \langle S, A, P^a_{ss'}, R^a_{ss'} \rangle$ is defined by a set of states S, a set of actions A, a transition model $P^a_{ss'}$ specifying the distribution over future states s' when an action a is performed in state s, and a corresponding reward model $R^a_{ss'}$ specifying a scalar cost or reward. A state can be a discrete atomic entity, such as a number; a factored object such as a vector of real state variables; or a structured object defined by a set of predicates or relations. Abstractly, a value function is a mapping $S \rightarrow \mathbb{R}$, or equivalently a vector $\in \mathbb{R}^{|S|}$ (when the state space is discrete). A deterministic policy $\pi : S \rightarrow A$ is a functional mapping from states to actions, whereas a stochastic policy induces a distribution over actions. Any policy induces a value function V^π, specifying the expected long-term discounted sum of rewards received by the agent

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

in any given state s when actions are chosen using the policy. Any optimal policy π^* defines the same unique optimal value function V^*, which satisfies the Bellman equation:

$$V^*(s) = \max_a \sum_{s'} P^a_{ss'} \left(R^a_{ss'} + \gamma V^*(s') \right).$$

Classical techniques for solving MDPs include *value iteration* and *policy iteration* (Puterman 1994). In small MDPs, it is possible to store value functions exactly as a table. Larger problems require the use of a function approximator to generalize estimated values across the state space. Temporal-difference learning (TD) (Sutton and Barto 1998) has been shown to be an effective sampling-based method for solving large MDPs, when combined with a suitable function approximator. Nonlinear approximation methods such as neural nets have led to significant empirical successes with TD (Tesauro 1992), but suffer from convergence problems (Bertsekas and Tsitsiklis 1996). More recent methods, such as least-squares policy iteration (Lagoudakis and Parr 2003) and linear programming (Guestrin et al. 2003), are based on a *linear* function approximation architecture using a set of $k \ll |S|$ *handcoded* basis functions $\{\phi_1, \ldots, \phi_k\}$, such as orthogonal polynomials (Lagoudakis and Parr 2003) and radial basis functions (RBFs). Hierarchical RL methods are based on *semi-MDPs* (SMDPs), which allow temporally extended actions like “exiting a room” or “driving home”, which correspond to executing a hierarchical policy over a portion of the state space (Barto and Mahadevan 2003). The MDP framework has also been extended to richer state descriptions using first-order representations (FOMDPs) (Boutilier, Reiter, and Price 2001). *Partially observable MDPs* (or POMDPs) address the problem of acting when the underlying state is hidden (Kaelbling, Littman, and Cassandra 1998).

Representation Discovery Problems

I discuss three representation discovery problems in MDPs: finding functional abstractions, state space abstractions, and discovering temporal abstractions.

Functional Abstraction

One general class of algorithms for constructing novel representations is to find task-dependent basis functions defined on the state (action) space that span linear subspaces containing the optimal solution. This approach can be viewed as compressing the space of all (value) functions $f \in \mathbb{R}^{|S|}$ into those that can be represented as a linear combination of basis functions

$$f = \sum_{i \in I} w_i \phi_i$$

where I is a set of indices specifying the selected bases. The set of basis functions can be grouped together to form a matrix Φ of size $|S|$ by k, where each column corresponds to a basis function ϕ_i, and $k \ll |S|$. Each row of this matrix defines a set of features $\phi(s) \in \mathbb{R}^k$, which can be viewed as a real-valued vector embedding of the original state. The main idea here is to exploit problem-specific information in constructing basis functions. Two general approaches to constructing basis functions include reward-sensitive bases such as Bellman error basis functions (BEBFs) (Keller, Mannor, and Precup 2006; Parr et al. 2007) and Drazin bases (Mahadevan 2009) vs. reward-invariant bases such as proto-value functions (PVFs) (Mahadevan and Maggioni 2007) and geodesic Gaussian kernels (Sugiyama et al. 2008). Reward-invariant methods exploit the intuition that value functions are generally *smooth* functions on the state space, and can be represented sparsely using a small set of carefully constructed bases. Reward-sensitive methods construct bases that exploit specific knowledge of the task, and usually the policy as well.

State Abstraction

State abstraction methods induce a discrete mapping by partitioning the state (action) space into equivalence classes on which the value function assumes a constant value. State partitioning methods can be viewed as a special case of functional abstraction defined by a basis matrix Φ of size $|S|$ by k, where the embedding of a state $\phi(s) \in \mathbb{R}^k$ is a binary row vector with exactly a single 1 indicating the unique partition containing the state. Two states s and s' are considered equivalent by a state abstraction method if and only if $\phi(s) = \phi(s')$. The set of all state abstraction methods forms a partially ordered hierarchy, which can be organized into a *lattice* structure (Li, Walsh, and Littman 2006), based on the coarseness of the induced partition. Partitioning methods can be categorized into several classes: those that are model-preserving in that they respect the reward function and the transition dynamics (Givan and Dean 1997; Ravindran and Barto 2003); those that preserve the (action) value function for all (or only optimal) policies; and finally, those that preserve the optimal action. Pattern databases in deterministic search problems can be viewed as a special type of *additive state abstraction* that yield admissible heuristics (Yang et al. 2008). A linear programming based approach to learning admissible heuristics by feature discovery is described in (Petrik and Zilberstein 2008).

Temporal Abstraction

Finally, a third class of representation discovery methods for MDPs is based on learning temporal abstractions. One approach is to learn an abstraction hierarchy over the set of state variables, based on frequency of change (Hengst 2002), or causal dependencies among state variables (Jonsson and Barto 2005). Another approach is to learn reusable policy fragments or *skills*, based on finding bottlenecks (Şimşek and Barto 2004). Recent work on discovery of temporal hierarchies in POMDPs is based on nonlinear quadratic programming (Charlin, Poupart, and Shioda 2007).

Representation Discovery Algorithms

In this section, I categorize algorithms for representation discovery into four categories: whether representations are constructed by dilation or diagonalization; whether representations are reward-sensitive or reward-independent; whether flat or multiscale representations are constructed; and finally, whether a minimal or overcomplete set of representations is constructed. These categories not intended to be exhaustive, however, but rather reflect recent work in the field.

Dilation vs. Diagonalization Methods

A general strategy for constructing representations is based on diagonalization or dilation of an exact or approximate transition model. *Krylov* methods (Petrik 2007; Poupart and Boutilier 2003) are based on dilating the reward function using powers of the transition matrix P^π. The Krylov space is the smallest subspace invariant under the reward function R^π and transition matrix P^π, which can be constructed by orthogonalizing the vectors

$$\{R^\pi, P^\pi R^\pi, (P^\pi)^2 R^\pi, \ldots, (P^\pi)^{n-1} R^\pi\}.$$

BEBF representations are an incremental variant of Krylov bases, which use the (sampled) Bellman error as basis vectors. The concept of dilation can be generalized to first-order MDPs, where it is referred to as the *regression* of a reward function over a specific action (Boutilier, Reiter, and Price 2001).

Diagonalization methods are based on finding the eigenvectors ϕ_i of a transition matrix P^π, where $P^\pi \phi_i = \lambda_i \phi_i$. Since the diagonalization of arbitrary transition matrices may not yield real-valued or orthogonal eigenvectors, it is often expedient to use a reversible approximation $\hat{P}^\pi$, such as the natural random walk on a state space graph induced by the policy π, where two states i and j are connected by an undirected edge if there exists some action a such that $P^a_{i,j} > 0$ or $P^a_{j,i} > 0$ (Mahadevan and Maggioni 2007). In this case, the reversible random walk stochastic matrix is defined by $\hat{P}^\pi = D^{-1}W$, where W is the induced connectivity matrix and D is a diagonal matrix of its row sums. It is more tractable to use the spectrally similar symmetric "normalized" Laplacian matrix $\mathcal{L} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$, an object of much recent study in machine learning.

Reward-Specific vs. Reward-Invariant Approaches

Reward-respecting state abstraction methods consider two states s and s' equivalent (or $\phi(s) = \phi(s')$) if and only if $R(s,a) = R(s',a)$ under each action a. Similarly, reward-sensitive basis construction methods, like Krylov bases, use knowledge of the reward function to find representations of subspaces that contain the (approximately) optimal value function. Reward-invariant approaches, such as PVFs or geodesic Gaussian kernels, construct basis functions reusable across multiple reward functions on the same state (action) space. Petrik (Petrik 2007) proposed combining reward-specific Krylov bases with reward-invariant proto-value functions as a way of integrating localized reward-specific and more global eigenvector representations.

Flat vs. Multiscale Methods

Multiscale methods construct a variable resolution spatial or temporal abstraction hierarchy. One approach uses a hierarchical matrix approximation method called diffusion wavelets (Coifman and Maggioni 2006), which constructs a sparse representation of dyadic powers of a transition matrix. Similarly, Hengst (Hengst 2002) constructs a multi-scale state hierarchy by partitioning the state variables based on their frequency of change. In contrast, flat methods construct a single-level abstraction, such as eigenvector methods like proto-value functions, or Krylov bases.

Orthogonal vs. Redundant Representations

Representation discovery methods that construct orthogonal representations can be contrasted with those that construct redundant representations. Orthonormal bases, such as proto-value functions, represent value functions (or reward functions) uniquely as a weighted linear combination of basis elements, where the weighting is given by $\langle V^\pi, \phi_i \rangle$, the projection of V^π onto the i^{th} basis element:

$$V^\pi = \sum_{i \in I} \langle V^\pi, \phi_i \rangle \phi_i.$$

Overcomplete basis representations, such as diffusion wavelets, can represent a given value function in many different ways, in which case an additional *regularization* step can be used to find a *sparse* representation (Johns and Mahadevan 2009; Kolter and Ng 2009).

Learning Representation and Control

One framework for combining the learning of representation and control is called Representation Policy Iteration (RPI) (Mahadevan and Maggioni 2007), where the outer loop finds a functional abstraction based on a specific policy (or reward function), and the inner policy evaluation loop finds the closest (least-squares regularized) approximation within the span of the constructed bases. An alternative approach is to construct representations during the control learning phase itself (Parr et al. 2007). The space of such hybrid representation-and-control learning architectures needs to be explored further.

Challenges

One significant challenge is how to scale representation discovery algorithms to high-dimensional problems. Some directions include low-rank matrix approximation (Johns, Mahadevan, and Wang 2007), exploiting pre-defined task hierarchies (Osentoski and Mahadevan 2010), and using relational representations (Wu and Givan 2007). The scalability of these methods to much larger discrete MDPs, such as Tetris (Bertsekas and Tsitsiklis 1996) or backgammon (Tesauro 1992), and continuous state and action MDPs, such as helicopter control (Ng et al. 2004), needs to be explored further. Much of the work on representation discovery has been in fully observable MDPs. Research on constructing novel representations in POMDPs is ongoing (Poupart and Boutilier 2003; Li et al. 2007; Charlin, Poupart, and Shioda 2007). Finally, constructing representations that transfer across tasks remains an important challenge (Taylor, Kuhlmann, and Stone 2008).

Acknowledgements

This research was supported in part by the National Science Foundation under grants NSF IIS-0534999 and NSF IIS-0803288. I thank Jeff Johns, George Konidaris, and the anonoymous reviewers for their feedback on this paper.

References

Amarel, S. 1968. On representations of problems of reasoning about actions. In Michie, D., ed., *Machine Intelligence 3*, volume 3, 131–171. Elsevier/North-Holland.

Barto, A., and Mahadevan, S. 2003. Recent advances in hierarchical reinforcement learning. *Discrete Event Systems Journal* 13:41–77.

Bertsekas, D., and Tsitsiklis, J. 1996. *Neuro-Dynamic Programming*. Belmont, Massachusetts: Athena Scientific.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11:1–94.

Boutilier, C.; Reiter, R.; and Price, B. 2001. Symbolic dynamic programming for first-order MDPs. In *IJCAI'01: Proceedings of the 17th international joint conference on Artificial intelligence*, 690–697. Morgan Kaufmann Publishers Inc.

Charlin, L.; Poupart, P.; and Shioda, R. 2007. Automated hierarchy discovery for planning in partially observable environments. In Schölkopf, B.; Platt, J.; and Hofmann, T., eds., *Advances in Neural Information Processing Systems 19*, 225–232.

Coifman, R., and Maggioni, M. 2006. Diffusion wavelets. *Applied and Computational Harmonic Analysis* 21(1):53–94.

Connell, J., and Mahadevan, S. 1993. *Robot Learning*. Kluwer Academic Press.

Givan, R., and Dean, T. 1997. Model Minimization in Markov Decision Processes. In *Proceedings of the AAAI*.

Guestrin, C.; Koller, D.; Parr, R.; and Venkataraman, S. 2003. Efficient solution algorithms for factored MDPs. *Journal of AI Research* 19:399–468.

Hengst, B. 2002. Discovering hierarchy in reinforcement learning with HEXQ. In *ICML*, 243–250.

Johns, J., and Mahadevan, S. 2009. Sparse approximate policy evaluation using graph-based basis functions. Technical Report UM-CS-2009-41, Department of Computer Science, University of Massachusetts Amherst.

Johns, J.; Mahadevan, S.; and Wang, C. 2007. Compact spectral bases for value function approximation using Kronecker factorization. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Jonsson, A., and Barto, A. 2005. A causal approach to hierarchical decomposition of factored MDPs. In *Proceedings of the 22nd international conference on Machine learning*, 401–408. New York, NY, USA: ACM.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101:99–134.

Keller, P.; Mannor, S.; and Precup, D. 2006. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *Proceedings of the* 22^{nd} *International Conference on Machine Learning (ICML)*, 449–456. MIT Press.

Kolter, J. Z., and Ng, A. Y. 2009. Regularization and feature selection in least-squares temporal difference learning. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, 521–528.

Lagoudakis, M., and Parr, R. 2003. Least-squares policy iteration. *Journal of Machine Learning Research* 4:1107–1149.

Li, X.; Cheung, W. K. W.; Liu, J.; and Wu, Z. 2007. A novel orthogonal NMF-based belief compression for POMDPs. In *Proceedings of the 24th international conference on Machine learning*, 537–544.

Li, L.; Walsh, T.; and Littman, M. 2006. Towards a unified theory of state abstraction for MDPs. In *In Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics*, 531–539.

Mahadevan, S., and Maggioni, M. 2007. Proto-Value Functions: A Laplacian Framework for Learning Representation and Control in Markov Decision Processes. *Journal of Machine Learning Research* 8:2169–2231.

Mahadevan, S. 2009. Learning Representation and Control in Markov Decision Processes: New Frontiers. *Foundations and Trends in Machine Learning* 1(4):403–565.

Ng, A.; Kim, H.; Jordan, M.; and Sastry, S. 2004. Autonomous helicopter flight via reinforcement learning. In *Proceedings of Neural Information Processing Systems*.

Osentoski, S., and Mahadevan, S. 2010. Basis function construction in hierarchical reinforcement learning. In *AAMAS '10: Proceedings of the 9th international joint conference on Autonomous agents and multiagent systems*.

Parr, R.; Painter-Wakefiled, C.; Li, L.; and Littman, M. 2007. Analyzing feature generation for value function approximation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 737–744.

Petrik, M., and Zilberstein, S. 2008. Learning heuristic functions through approximate linear programming. In *International Conference on Automated Planning and Scheduling (ICAPS*, 248–255.

Petrik, M. 2007. An analysis of Laplacian methods for value function approximation in MDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2574–2579.

Poupart, P., and Boutilier, C. 2003. Value directed compression of POMDPs. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*.

Puterman, M. L. 1994. *Markov Decision Processes*. New York, USA: Wiley Interscience.

Ravindran, B., and Barto, A. 2003. SMDP homomorphisms: An algebraic approach to abstraction in Semi-Markov Decision Processes. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*.

Şimşek, O., and Barto, A. 2004. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*. New York, NY, USA: ACM.

Sugiyama, M.; Hachiya, H.; Towell, C.; and Vijayakumar, S. 2008. Geodesic gaussian kernels for value function approximation. *Autonomous Robots* 25:287–304.

Sutton, R., and Barto, A. G. 1998. *An Introduction to Reinforcement Learning*. MIT Press.

Taylor, M. E.; Kuhlmann, G.; and Stone, P. 2008. Autonomous transfer for reinforcement learning. In *AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, 283–290.

Tesauro, G. 1992. Practical issues in temporal difference learning. *Machine Learning* 8:257–278.

Wu, J.-H., and Givan, R. 2007. Discovering relational domain features for probabilistic planning. In *ICAPS*, 344–351.

Yang, F.; Culberson, J.; Holte, R.; Zahavi, U.; and Felner, A. 2008. A general theory of additive state space abstractions. *J. Artif. Int. Res.* 32(1):631–662.