Learning with Deictic Representations

Leslie Pack Kaelbling, Sarah Finney, Natalia Gardiol & Tim Oates

Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

http://www.ai.mit.edu

The Problem: Humans speak, and probably think, of the world as being made up of objects. Individual objects have features, and sets of objects stand in various relations to one another. If artificial agents are to successfully interact with our world, they will need to represent and reason about objects, their features, and relations that hold among them. We are interested in how agents embedded in inherently relational environments, such as physically embodied robots, can learn to act.

What learning algorithms are appropriate? What representations are appropriate? The answers to these questions interact. First order logic (FOL) immediately comes to mind as a possible representation due to its compactness and ability to generalize over objects via quantification. However, reinforcement learning, one of the most widely studied methods for learning how to act in complex, stochastic domains, is not well-suited for logical representations. Rather, reinforcement learning algorithms typically require propositional representations, which do not have the generalization properties of first order representations and tend to be much larger. Is there any middle ground?

Previous Work: One strategy that has been successful in AI planning, for example, is to propositionalize relational domains. That is, to specify a (hopefully small) finite domain of objects, and to describe the domain using a large set of propositions representing every possible instantiation of the properties and relations in the domain. This method may be inappropriate for learning because it does not give any representational support for generalization over objects.

Starting with the work of Agre and Chapman [1], and gaining momentum with the debate over the fruitcake problem in the late 1980's (see the Winter, 1989 issue of AI Magazine for the culmination of this debate), arguments have been made for the viability of deictic representations in relational domains. Such representations point to objects in the world and name them according to their role in ongoing activity rather than with arbitrary identifiers, as is typically the case with first order representations. Despite the fact that deictic representations lack quantification, they allow for some of the generalization that FOL affords without the attendant computational complexity. Perhaps most importantly, they can be used with some existing reinforcement learning algorithms.

Even though deictic representations appear to be attractive, there has been no systematic investigation of the circumstances under which they result in better performance compared to naive propositional representations when the underlying domain is relational.

Approach: The term *reinforcement learning* (RL) is used to denote both a set of problems and a set of algorithms. Reinforcement learning problems are framed in terms of an agent, an environment, and rewards. The agent can observe the environment and can affect the environment by taking actions. Each time the agent takes an action it receives a scalar reward signal that indicates how good it was to have taken that particular action given the current state of the environment. The task facing the agent is to learn a mapping from observations (or histories of observations) to actions that maximizes the reward that it receives over time.

One way of representing states is atomically, with one atom for each unique configuration of the blocks. It might then be possible to learn, for example, that taking action a_2 in state 2397 yields a reward of 0.5 and leads to state 78512. This representation is simple in the sense that a single number can represent an entire state and learning can be implemented using tables indexed by states. The price to be paid for this simplicity is high. First, atomic representations can be intractably large, even for small numbers of objects. Given n blocks, there are 3^n possible assignments of colors to blocks and $O(n!)$ ways to stack the blocks. Second, because there is no way to determine whether two states are similar, there is no way to generalize to learn, for example, that if action a_2 is the best one to take in state 2397 then it is probably also the best one to take in state 78512.

The term *propositional representation* is to denote a naive propositionalization of a relational domain. For example, given n blocks there would be $3n$ propositions to specify the colors of the blocks and n^2 propositions to specify how the blocks are stacked.

Though propositional representations afford some opportunities for generalization, they must refer to objects by name (e.g., *block12* and *block8*) and are therefore unable to perform an important type of generalization. Deictic representations have the potential to bridge this gap, allowing much of the generalization afforded by FOL representations yet being amenable to solution (even in the face of uncertainty) by existing algorithms.

The word deictic derives from the Greek *deiktikos*, which means "able to show". It is used in linguistics, and was introduced into the artificial intelligence vernacular by Agre and Chapman [1], who were building on Ullman's work on visual routines [6]. A deictic expression is one that "points" to something; its meaning is relative to the agent that uses it and the context in which it is used. "The book that I am holding" and "the door that is in front of me" are examples of deictic expressions in natural language.

Our learning agent exists in a simulated blocks-world and must learn to use its hand to remove any red or blue blocks on a green block so that block may be lifted. The choice of this problem domain was not arbitrary. Whitehead and Ballard [7] introduced it in their pioneering work on the use of deixis in relational domains. They developed the Lion algorithm to deal with the *perceptual aliasing* (partial observability) that resulted from using a deictic representation. McCallum [5] used the same domain to demonstrate that the number of markers required to solve the problem can be reduced by keeping a short history of observations. Finally, Martin [4] used the domain to motivate and evaluate an algorithm for learning policies that generalize over initial configurations.

While the choice of a deictic representation gives us an observation space that does not get larger as the world becomes more complex (within the blocks-world domain), we now have to include some history in order to make the problem Markov. Thus, the observation space is still too large for an explicit representation of the value function, such as a table. We chose two classes of value function approximation algorithms: neuro-dynamic programming [2] (NDP), and tree-based selective-perception reinforcment learning algorthms, the G algorithm [3] and McCallum's UTree algorithm [5]. In neuro-dynamic programming, neural networks are used to approximate the Q-value of each state-action pair. G and UTree both look at reward distributions to determine which observation bits are relevant to predicting reward and divide the state space up accordingly. SARSA(λ) is then used to learn Q-values for these subdivided states.

Impact: Propositional representations yield large observation spaces and full observability. Deictic representations yield small observation spaces and partial observability. Which makes learning easier? That is the question that we seek to understand. We are investigating the trade-offs, some unexpected, between deictic and propositional representations. We are also investigating issues arising from the use of reward distributions to divide the state space.

References:

[1] Philip E. Agre and David Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, volume 1, pages 268–272, Seattle, Washington, 1987. Morgan Kaufmann.

[2] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, Massachusetts, 1996.

[3] David Chapman and Leslie Pack Kaelbling. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Sydney, Australia, 1991.

[4] Mario Martin. *Reinforcement Learning for Embedded Agents facing Complex Tasks*. PhD thesis, Universitat Politecnica de Catalunya, Barcelona, Spain, 1998.

[5] R. Andrew McCallum. Instance-based utile distinctions for reinforcement learning with hidden state. In *Proceedings of the Twelfth International Conference Machine Learning*, pages 387–395, San Francisco, CA, 1995. Morgan Kaufmann.

[6] Shimon Ullman. Visual routines. *Cognition*, 18:97–159, 1984.

[7] Steven D. Whitehead and Dana H. Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83, 1991.