Decision Tree Methods For Finding Reusable MDP Homomorphisms

Alicia Peregrin Wolfe and Andrew G. Barto

Autonomous Learning Laboratory
University of Massachusetts, Amherst
pippin@cs.umass.edu, barto@cs.umass.edu

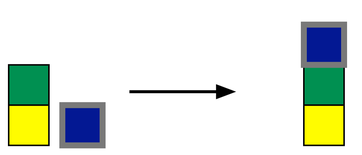
Abstract

State abstraction is a useful tool for agents interacting with complex environments. Good state abstractions are compact, reuseable, and easy to learn from sample data. This paper combines and extends two existing classes of state abstraction methods to achieve these criteria. MDP homomorphisms (Ravindran, 2004), produce models of reward and transition probabilities in an abstract state space. The UTree algorithm (McCallum, 1995) learns compact models of the value function quickly from sample data but builds models which cannot be reused. We present results showing a new, combined algorithm that fulfills all three criteria: the resulting models are compact, can be learned quickly from sample data, and can be used across a class of reward functions.

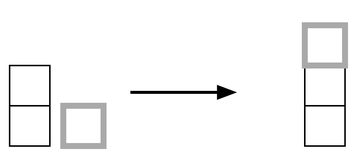
Example

One abstraction for multiple related tasks: tasks which depend on the same variable.

Moving a block in a blocks world to different positions:



True State Transition



Abstract State Transition

Reuseable Homomorphisms

Homomorphism: Mapping to abstract states/actions, preserves some aspect of the original model, $h: S \times A \rightarrow S' \times A'$.
Consists of state and action mappings

- $f: S \rightarrow S'$
- $g_s: A \rightarrow A'$

CMP Homomorphisms

Output variable y
- object position, size, color
- level of light in a room

Preserve two properties:
- Immediate prediction of the output variable

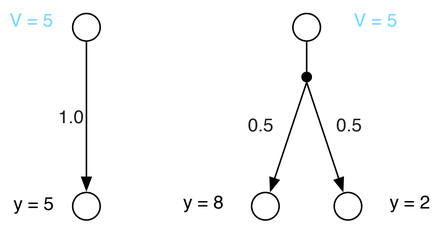
  $y'(f(s)) = y(s)$
- Prediction of the abstract next state

  $T'(f(s_i), g_{s_i}(a), f(s_j)) = \Sigma_{s_k \mid f(s_k) = f(s_j)} \; T(s_i, a, s_k)$

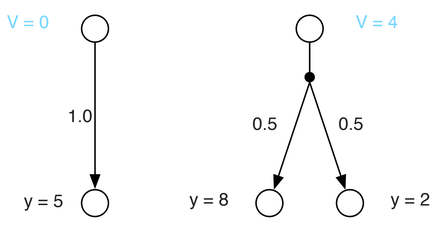
Based closely on MDP Homomorphisms (Ravindran, 2004)

Value Function Model

Most existing methods of finding abstract MDP models from data focus on finding a model which predicts only the optimal Value Function for a single task. Most of these methods are similar to the Utree algorithm of McCallum (1995).



r o y = y

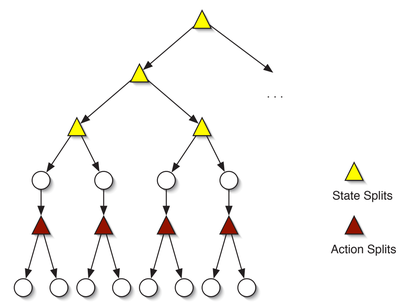


r o y = y if y >5, 0 otherwise

However, the algorithms can be modified to define an algorithm that does find CMP Homomorphisms.

Algorithm

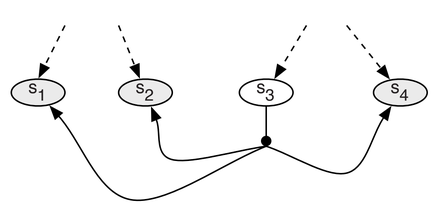
The algorithm builds a decision tree based on features in order to avoid enumerating every state and action in the MDP.
- Splits: state and action features
- State abstraction leaves: *f(s)*
- Action abstraction leaves: $g_s(a)$

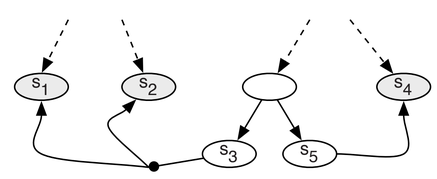


Splitting Leaves

Use the state abstraction defined by the state leaves (*f(s)*) to calculate estimated abstract transition probabilites at each action leaf.

Split if feature improves either:
- Next abstract state prediction
- Immediate output prediction (*y*)





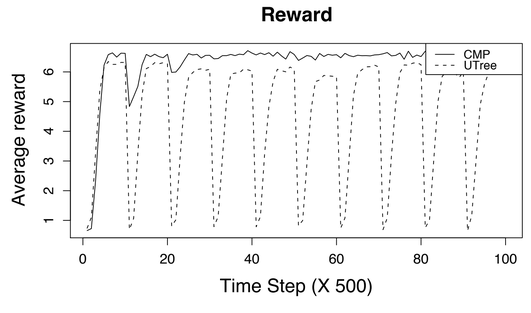
Results

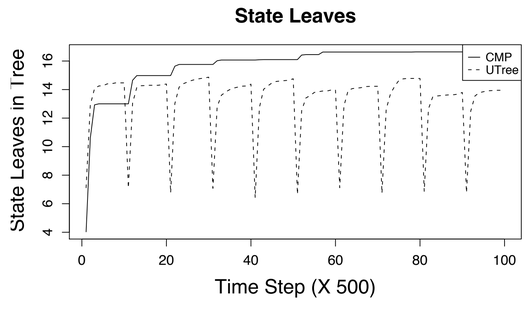
These experiments compare UTree, with its value function based abstraction, to the CMP Homomorphism tree.

Each experiment presents the agent with a series of related tasks (like setting the same bits in the counter to different values). The agent has a fixed time window in which to solve each task, then the next related task is presented. The goal is to have an agent which reuses information from earlier tasks to solve later ones.

Integer Counter Tasks

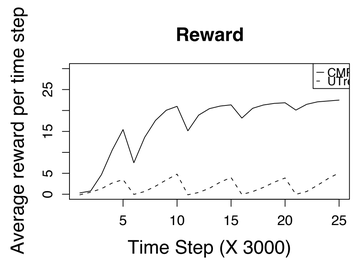
- 10 bit counter (2^{10} states)
- actions: add 1, subtract 1
- Task: achieve given setting of bits 3 & 4 (i.e. 10__ or 11__, etc)
- Task changes every 5000 time steps

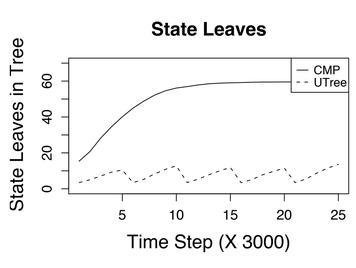




Blocks World Tasks

- Blocks world with 3 blocks
- Task: given a focus block, place it at location (pile x, height y)
- Task changes every 15000 time steps
- CMP Tree size and performance improve with each task
- UTree cannot find correct abstraction when task is hard (y = 3)





Acknowledgements

This research was facilitated in part by a National Physical Science Consortium Fellowship and by stipend support from Sandia National Laboratories, CA. This research was also funded in part by NSF grant CCF 0432143.

The full paper url: http://www-all.cs.umass.edu/~pippin/publications/AAAI0613WolfeA.pdf