Deciding to Specialize and Respecialize a Value Function for Relational Reinforcement Learning

UNIVERSITY OF MICHIGAN

Mitchell Keith Bloch and John Edwin Laird

The Soar Group at the University of Michigan

Big Idea

- We test relational reinforcement learning agents using all three of our specialization criteria (see right) as we add respecialization functionality
- It was Bloch and Laird's [2015] hypothesis that it would be more efficient to specialize the value function quickly, making potentially suboptimal specializations as a result, and to later modify that value function in the event that the agent finds it could have made significantly better specializations

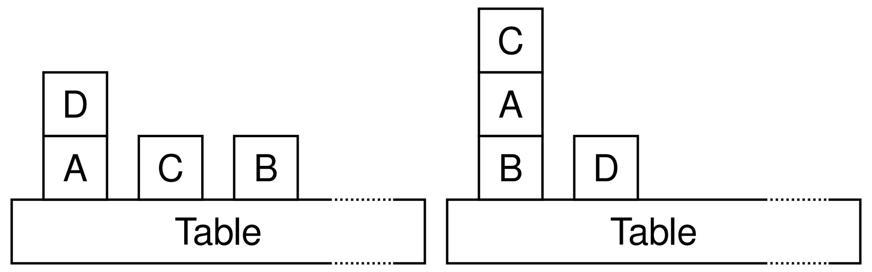
Reinforcement Learning

- Must learn how to act, given experience perceiving states, trying actions, and receiving rewards
- Explore with an ϵ-greedy exploration strategy
- Our agents learn using GQ(λ)
- At the most abstract:
  - $\pi(\boldsymbol{s}, \boldsymbol{a})$ represents the target policy
  - $\phi(\boldsymbol{i})$ represents the set of possible features
  - $\theta(\boldsymbol{i})$ stores weights which sum to provide value estimates for different actions

Relational RL

- Each state is described by a set of relations, such as `(<stack> ^top <block>)`
- Each feature in $\phi(\boldsymbol{i})$ represents a conjunction of any number of such relations
- Value function computation could dominate CPU time since variable bindings are expensive
- We use the Rete algorithm
  - It was designed for expert system rules
  - Handles variable bindings very efficiently
  - CPU time proportional to changes in environment rather than the total size of the environment
  - Shares work between similar rules

Our Relational Blocks World

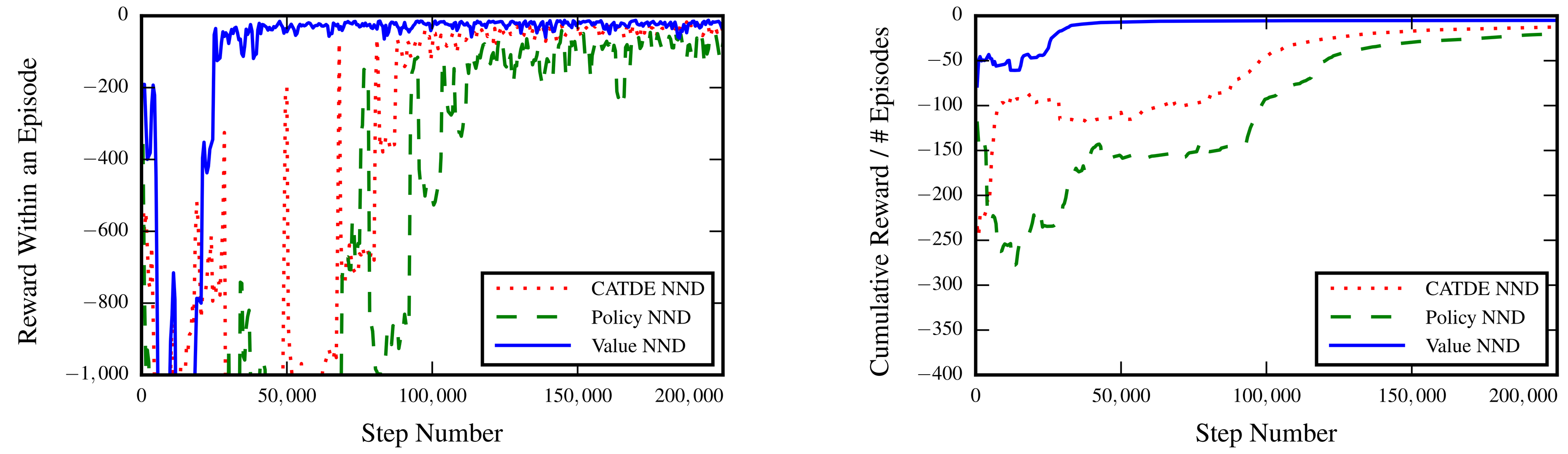


Blocks　　Goal

- Full representation of the goal presented by the environment
- Allows variable goals from episode to episode
- Allows variable numbers of blocks from episode to episode
- Complex training goal requires testing more than one relation

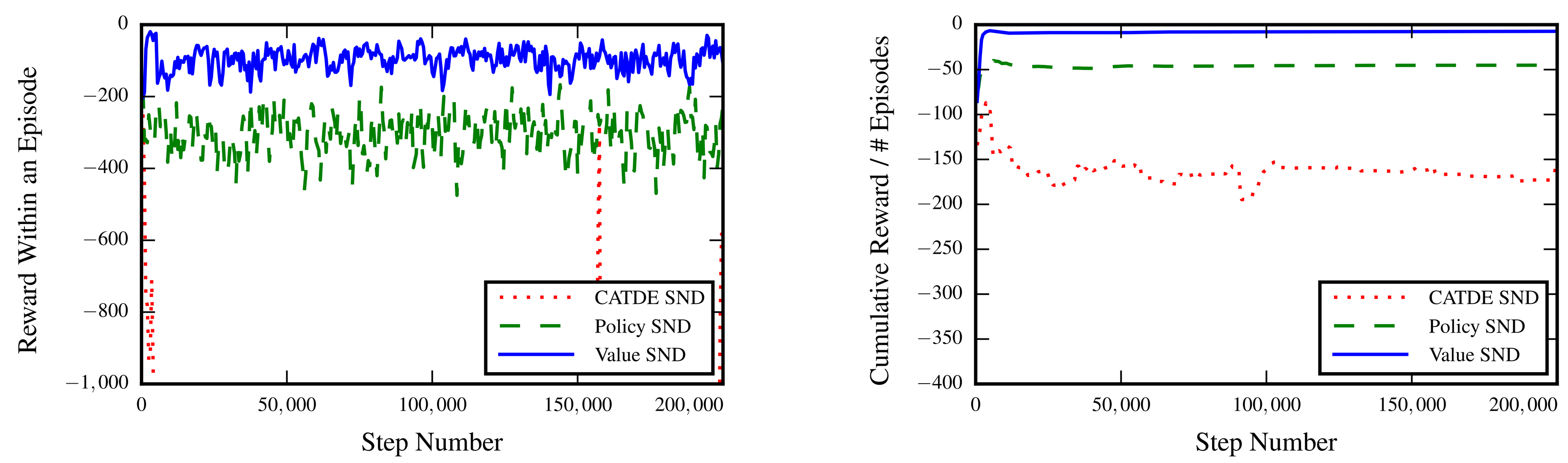
Agents Version #1: No Despecialization

- We start with agents which are capable of dynamically specializing the value function
- However, they cannot choose to undo seemingly suboptimal specializations in favor of potentially better ones



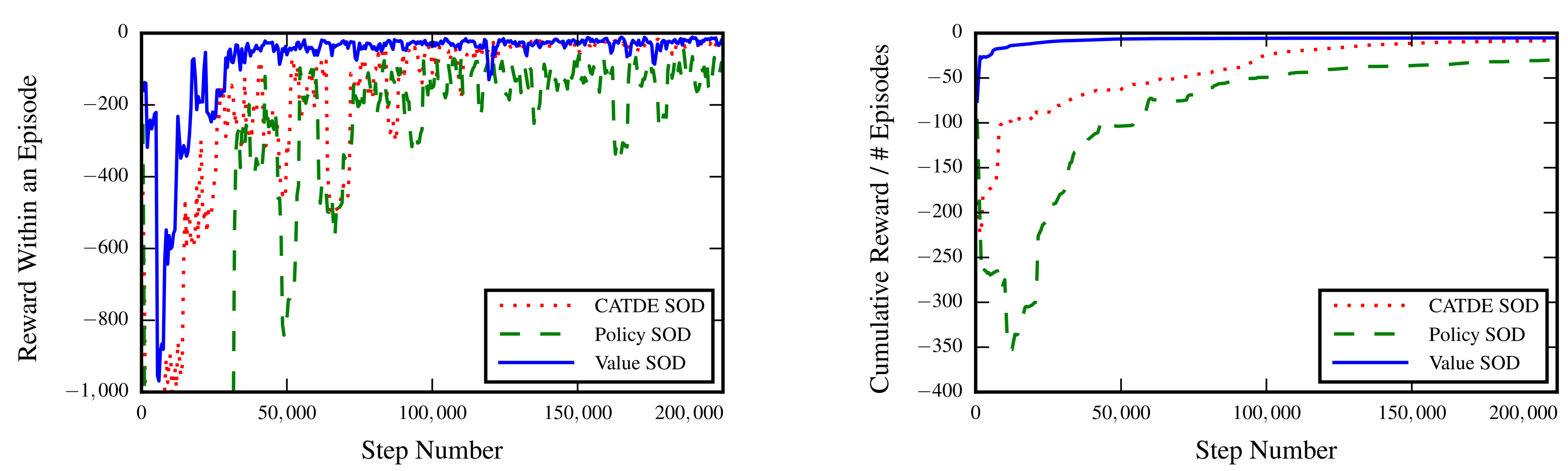
Agents Version #2: Respecialization

- We add the ability to respecialize
- However, we provide no mechanism to guarantee convergence on feature selection
- Thrashing results for two of our specialization criteria



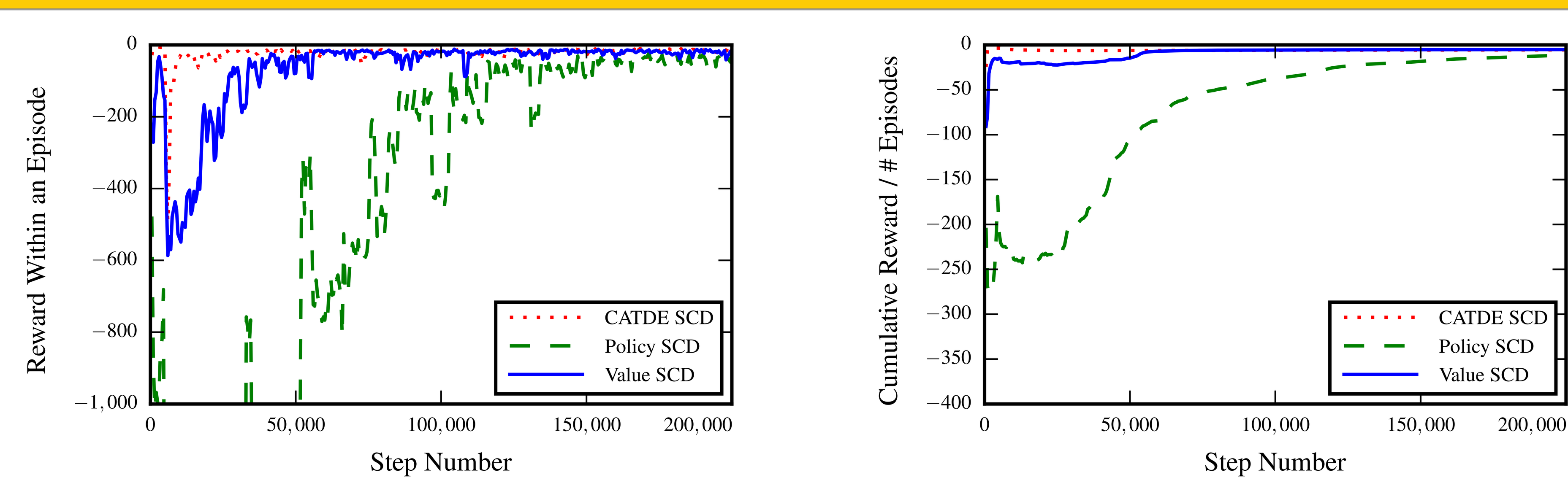
Agents Version #3: Respecialization + Boost

- We provide a mechanism to incrementally boost the likelihood of reselecting previously selected features
- Features chosen early are expected to be quite good, so eventully converging on one of them is likely to allow efficient learning
- While this guarantees eventual value function convergence, it does so at a significant computational cost



Agents Version #4: Respecialization + Boost + Concrete

- Freezing feature selection choices after a certain amount of evaluations without respecialization allows us to reduce computational load
- It additionally can result in a reduction in regret, even when compared to the agents using boost without the concrete mechanism



Final Cumulative Reward | Average Runtimes of Our Agents

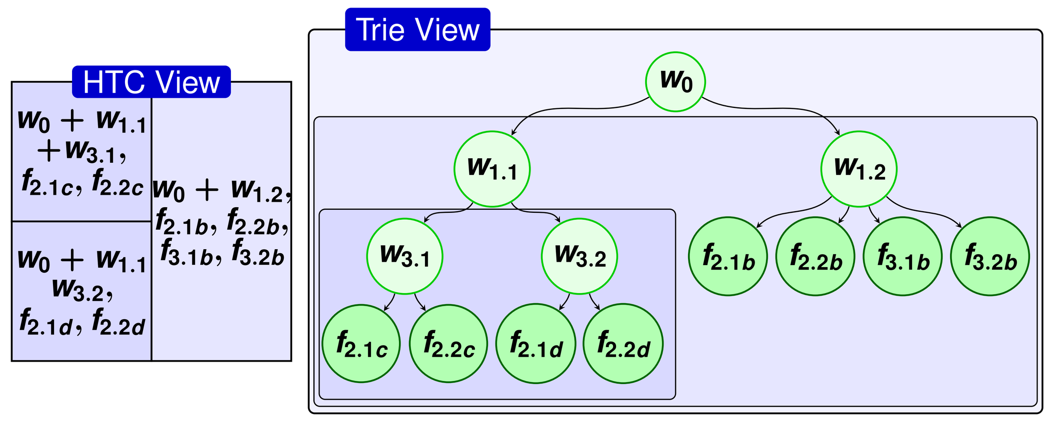
	No Respecialization	Respecialization	With Boost	With Boost+Concrete
CATDE	-12.5 | 13.8s	-158.3 | 27.4s	-8.3s | 44.8s	-5.5 | 11.0s
Policy	-20.2 | 18.0s	-45.1 | 1.2s	-30.0s | 96.6s	-11.5 | 11.0s
Value	-5.1 | 19.2s	-7.4 | 1.6s	-5.3s | 48.8s	-5.2 | 14.1s

Dynamic Specialization

- Given $\phi(\boldsymbol{i})$, $\theta(\boldsymbol{i})$, and other metadata, which features are most likely to improve the value function?
- We've explored the following criteria:
  - Cumulative Absolute Temporal Difference Error
  - Policy –Maximal change in $\pi(\boldsymbol{s}, \boldsymbol{a})$
  - Value – Maximal change in $\theta(\boldsymbol{i})$

Value Function Specialization

- Less refined tilings correspond to conjunctions of few features
- More refined tilings correspond to conjunctions of many features
- The most refined tilings correspond to fringe nodes i.e. candidate conjunctions for inclusion in the value function



Contributions

- This work implemented much of the future work suggested by Bloch and Laird [2015].
- We provide evidence to support Bloch and Laird's [2015] hypothesis that it is more efficient to specialize the value function quickly, making potentially suboptimal specializations as a result, and to later modify that value function in the event that the agent finds it could have made significantly better specializations

Future Work

- A higher order grammar for adding variables and new relations using these variables – for scenarios where an unbounded number of objects may be of interest
- More domains (suggestions welcome!)