Modular Reinforcement Learning in Soar

Shiwali Mohan and John Laird

Computer Science and Engineering
University of Michigan

June 16th, 2011

Outline

Soar Reinforcement Learning

- Framework
  - Rl-rules assign numeric preferences to *state-operator* pairs
    - Multiple rl-rules can match any given *state-operator* pair
    - Test for different features, combination of features
  - Value of the preferences is learned using *QLearning* or *SARSA*
  - Value of numeric preferences determines which operator is selected

Soar Reinforcement Learning

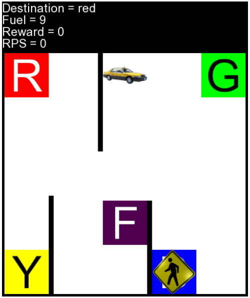
- Framework
  - Rl-rules assign numeric preferences to *state-operator* pairs
    - Multiple rl-rules can match any given *state-operator* pair
    - Test for different features, combination of features
  - Value of the preferences is learned using *QLearning* or *SARSA*
  - Value of numeric preferences determines which operator is selected
- Value Function
  - Distributed amongst rl-rules
  - Updates to value function is divided equally amongst all rl-rules

Soar Reinforcement Learning

- Framework
  - Rl-rules assign numeric preferences to *state-operator* pairs
    - Multiple rl-rules can match any given *state-operator* pair
    - Test for different features, combination of features
  - Value of the preferences is learned using *QLearning* or *SARSA*
  - Value of numeric preferences determines which operator is selected
- Value Function
  - Distributed amongst rl-rules
  - Updates to value function is divided equally amongst all rl-rules
- Action Selection
  - Numeric preferences with exploration

Complex Problems

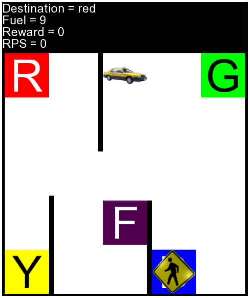
- Multiple goals and subgoals
- Hierarchical solutions
  - divide learning tasks into subtasks with termination conditions
  - subtasks can be combined sequentially to solve larger tasks
  - by Dietterich (2000)[1] for the *taxi-cab* domain

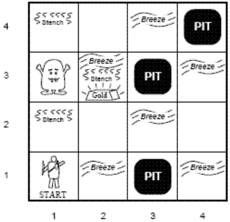


[1] Dietterich, T. G. (2000). Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition. *Journal of Artificial Intelligence Research*, 13(1):227–303

Complex Problems

- Multiple goals and subgoals
- Hierarchical solutions
  - divide learning tasks into subtasks with termination conditions
  - subtasks can be combined sequentially to solve larger tasks
  - by Dietterich (2000)[1] for the *taxi-cab* domain
- Not all problems can be divided into a series of subtasks
  - concurrent subtasks, interrupting
  - suggest contradicting actions
  - can only be partially satisfied





[1] Dietterich, T. G. (2000). Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition. *Journal of Artificial Intelligence Research*, 13(1):227–303

T-Maze Example

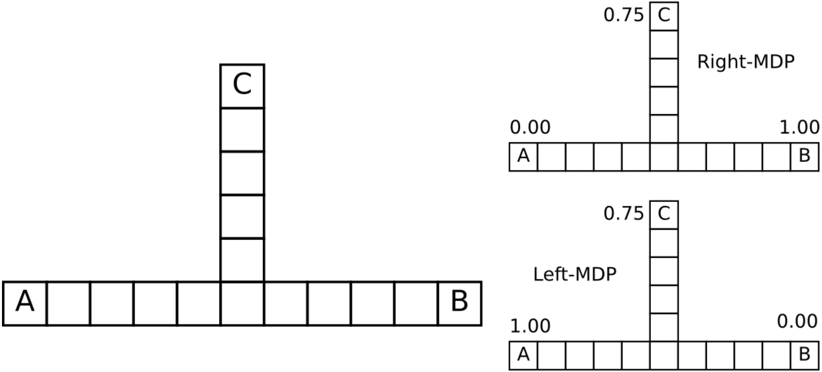


Figure: Multi-MDP T-Maze

The current Soar-RL framework can learn only the composite policy.

T-Maze Example

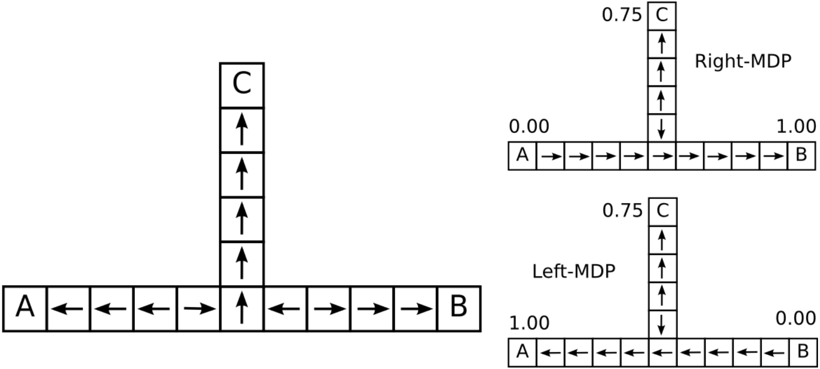


Figure: Multi-MDP T-Maze

The current Soar-RL framework can learn only the composite policy.

Modular Reinforcement Learning

- Goal
  - Discovering a composite policy for a set of N MDPs, $\{M_i\}_1^N$
  - Separate learning module (*sub-agent*) is created for each component MDP.

Modular Reinforcement Learning

- Goal
  - Discovering a composite policy for a set of N MDPs, $\{M_i\}_1^N$
  - Separate learning module (*sub-agent*) is created for each component MDP.
- Problem Formalization
  - as in Humphrys (1997)[1], Karlsson (1997)[2]
  - Each MDP has a distinct state space S_i
    - Composite state space $S = S_1 \times S_2 \times ... \times S_N$
  - Share a common action space, A
  - Each MDP has distinct reward R_i
    - Composite reward $R(s, a) = \sum_{i=1}^{N} R_i(s_i, a)$

[1] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[2] Karlsson, J. (1997). *Learning to Solve Multiple Goals*. PhD thesis, University of Rochester

Action Selection

- Multiple strategies can be used (Humphrys, 1997)[3]

[3] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[4] Bhat, S., Isbell, C., and Mateas, M. (2006). On the difficulty of Modular Reinforcement Learning for Real-World Partial Programming. In *In Proceedings of the the Twenty-First AAAI Conference on Artificial Intelligence*, volume 21, page 318. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

Action Selection

- Multiple strategies can be used (Humphrys, 1997)[3]
- Greatest Mass Learning
  - Actions are ordered by their summed Q-values, $X_a = \sum_j Q_j(s, a)$
  - Action selection may not be good for any single sub-agent

[3] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[4] Bhat, S., Isbell, C., and Mateas, M. (2006). On the difficulty of Modular Reinforcement Learning for Real-World Partial Programming. In *In Proceedings of the the Twenty-First AAAI Conference on Artificial Intelligence*, volume 21, page 318. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

Action Selection

- Multiple strategies can be used (Humphrys, 1997)[3]
- Greatest Mass Learning
  - Actions are ordered by their summed Q-values, $X_a = \sum_j Q_j(s, a)$
  - Action selection may not be good for any single sub-agent
- Top-Q Learning
  - Actions are ordered by their Top Q-values, $X_a = \max_j Q_j(s, a)$
  - Subagent with top Q-value may not have a strong preference amongst actions

[3] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[4] Bhat, S., Isbell, C., and Mateas, M. (2006). On the difficulty of Modular Reinforcement Learning for Real-World Partial Programming. In *In Proceedings of the the Twenty-First AAAI Conference on Artificial Intelligence*, volume 21, page 318. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

Action Selection

- Multiple strategies can be used (Humphrys, 1997)[3]
- Greatest Mass Learning
  - Actions are ordered by their summed Q-values, $X_a = \sum_j Q_j(s,a)$
  - Action selection may not be good for any single sub-agent
- Top-Q Learning
  - Actions are ordered by their Top Q-values, $X_a = \max_j Q_j(s,a)$
  - Subagent with top Q-value may not have a strong preference amongst actions
- Negotiated W-Learning
  - Select the subagent that stands to lose the most

[3] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[4] Bhat, S., Isbell, C., and Mateas, M. (2006). On the difficulty of Modular Reinforcement Learning for Real-World Partial Programming. In *In Proceedings of the the Twenty-First AAAI Conference on Artificial Intelligence*, volume 21, page 318. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

Action Selection

- Multiple strategies can be used (Humphrys, 1997)[3]
- Greatest Mass Learning
  - Actions are ordered by their summed Q-values, $X_a = \sum_j Q_j(s, a)$
  - Action selection may not be good for any single sub-agent
- Top-Q Learning
  - Actions are ordered by their Top Q-values, $X_a = \max_j Q_j(s, a)$
  - Subagent with top Q-value may not have a strong preference amongst actions
- Negotiated W-Learning
  - Select the subagent that stands to lose the most
- Impossible to attain *ideal* arbitration (Bhat et al., 2006)[4]
  - with properties like, *universality*, *unanimity*, *scale invariance*

[3] Humphrys, M. (1997). *Action Selection Methods Using Reinforcement Learning*. PhD thesis, University of Cambridge

[4] Bhat, S., Isbell, C., and Mateas, M. (2006). On the difficulty of Modular Reinforcement Learning for Real-World Partial Programming. In *In Proceedings of the the Twenty-First AAAI Conference on Artificial Intelligence*, volume 21, page 318. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

Changes in rules, input, structure

- Multiple rewards on the reward link
    - correspond to different MDPs present in the environment

```
...
(state <s> ^reward-link <rlink>)
(<rlink> ^reward_MDP1 <rmdp1>
         ^reward_MDP2 <rmdp2>)
(<rmdp1> ^reward.value <rvalue1>)
(<rmdp2> ^reward.value <rvalue2>)
...
```

Changes in rules, input, structure

- Multiple rewards on the reward link
  - correspond to different MDPs present in the environment

```
...
(state <s> ^reward-link <rlink>)
(<rlink> ^reward_MDP1 <rmdp1>
         ^reward_MDP2 <rmdp2>)
(<rmdp1> ^reward.value <rvalue1>)
(<rmdp2> ^reward.value <rvalue2>)
...
```

- RL-rules with *labels*

```
...
(state <s> ^feature1 <val1>
           ^feature2 <val2>
           ^operator <op>
           ^reward-link <rl>)
(<rl> ^reward_MDP1 <rmdp1>)
...
```

Changes in Algorithm

- Rl-rules and *labels* have a *many-to-many* ordering

Changes in Algorithm

- Rl-rules and *labels* have a *many-to-many* ordering
- Value function update is distributed by *labels*
  - discounted reward is divided equally amongst matched rl-rules with corresponding *label*
  - numeric value of an rl-rule is incremented by the sum of updates for different *labels* on it

Changes in Algorithm

- Rl-rules and *labels* have a *many-to-many* ordering
- Value function update is distributed by *labels*
  - discounted reward is divided equally amongst matched rl-rules with corresponding *label*
  - numeric value of an rl-rule is incremented by the sum of updates for different *labels* on it
- Action Selection
  - Current architecture supports *Greatest Mass Learning*
    - Operators are selected according to their combined numeric value $X_a = \sum_j Q_j(s, a)$ with some exploration
  - Other action selection schemes to be explored

T-Maze

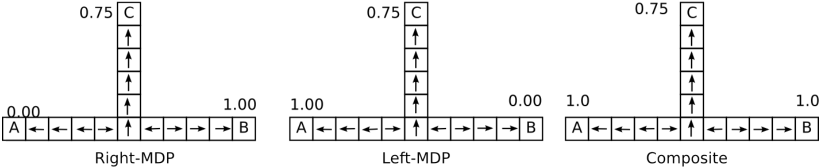


Figure: Soar-RL (10,000 runs)

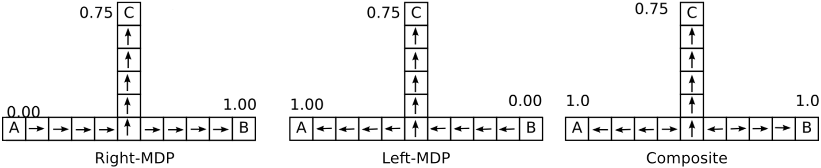


Figure: Soar-Modular-RL (10,000 runs)

Infinite Mario

- Object-oriented environment
- Previously,
  - Experimented with *action-selection* based on *class* of objects
  - Could not learn how to navigate in difficult situations
- Learn MDPs for a class of objects
  - Each object regulated a part of the reward signal
  - Individual updates to policy

Figure: Infinite Mario, difficulty 1

Infinite Mario - Results

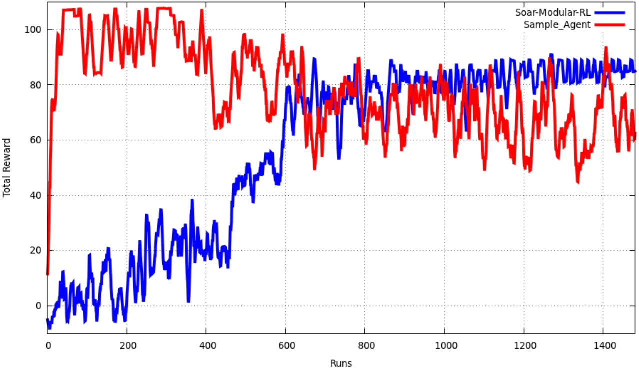


Figure: Infinite Mario, Difficulty 1, Seed 121

Conclusions

- Limitations
  - No optimality guarantee for the composite solution in QLearning
    - each module is guaranteed to converge to an optimal policy, value function
    - composite solution is guaranteed to converge (derives deterministically from component solutions)
  - Only very weak convergence guarantees in very specific situations for SARSA (Sprague, 2003)[5]

[5] Sprague, N. (2003). Multiple-goal Reinforcement Learning with Modular Sarsa (0). In *International Joint Conference on Artificial Intelligence*, number 0

Conclusions

- Limitations
  - No optimality guarantee for the composite solution in QLearning
    - each module is guaranteed to converge to an optimal policy, value function
    - composite solution is guaranteed to converge (derives deterministically from component solutions)
  - Only very weak convergence guarantees in very specific situations for SARSA (Sprague, 2003)[5]
- Future Work
  - More experiments with action selection
  - How can rewards be distributed amongst components?

[5] Sprague, N. (2003). Multiple-goal Reinforcement Learning with Modular Sarsa (0). In *International Joint Conference on Artificial Intelligence*, number 0

Nuggets and Coal

- Nuggets
  - An interesting, new approach to look at complex environments and faster solutions
  - A better solution to Infinite Mario problem (took ~3 years)
  - Can lead to better understanding of how soar-rules map to MDPs in RL setup
- Coal
  - Limited in variety of ways
  - Have done only limited experimentation with established methods
  - Distributing rewards amongst components might be a hard problem
  - Impossibility of an *ideal* arbitration function