Discovery, Evaluation, and Exploration of Human Supplied Options and Constraints

(Extended Abstract)

Jesse Rosalia*, Güliz Tokadlı**, Charles L. Isbell Jr.*, Andrea Thomaz* and Karen M. Feigh**

*College of Computing, Georgia Institute of Technology
801 Atlantic Dr. Atlanta, GA 30318
jesse.rosalia@gatech.edu, {isbell, athomaz}@cc.gatech.edu
**College of Engineering, Georgia Institute of Technology
270 Ferst Dr. Atlanta, GA 30332
{glztokadli, karen.feigh}@gatech.edu

Categories and Subject Descriptors:
Computing Methodologies [**Machine Learning**]: Learning Paradigms—*Reinforcement Learning*

Keywords: reinforcement learning; human input; work domain analysis; abstraction hierarchy; options; constraints

1. INTRODUCTION

This work focuses on soliciting human input to provide useful abstractions to a learning agent. It is an instance of both *interactive machine learning* and *cognitive engineering*. In particular, we use *Work Domain Analysis* (WDA) developed from a user study with a method called Abstraction Hierarchy to describe the high-level goals, strategies, and the action abstractions used by Pacman players of different skill levels [2]. This process yields both "should do" actions and "don't do" actions for each of the groups of users, high performers (HP) and low performers (LP). We implement a Q-learning agent for each group and compare the performance of each to the other, to an agent using only primitive actions, and to an agent using actions defined by machine learning experts. Unfortunately, human input can be noisy, with respect to specifying useful constraints; we show cases in which selectively defying a noisy constraint yields better performance than omitting the constraint, and introduce an algorithm for exploring and identifying such constraints.

2. EXPERIMENTAL SETUP

We implemented "should do" actions as options (i.e. subpolicies over primitive actions as defined in [1]), and we implemented each "don't do" action as the set $c \subset P(S \times A)$ that represents primitive actions that an agent cannot take in a given state. We call these sets "constraints." We observed that the constraints indicated by the participants were not used all the time during their gameplay; we believe a common characteristic of human supplied constraints is that there will exist important exceptions, when a constraint should not be followed. To evaluate this, we introduce a new exploration policy in which the agent is allowed to select constrained actions (i.e. "defy the constraint") with probability $1 - \upsilon$.

Each trial was run for 10000 training episodes on each of the three Pac-man boards (small, medium, and original). We sampled the policy of each agent at episode 1, 2, 5, 10, 25, 50, and then every 50 episodes afterward. These data were then averaged across all 200 trials. All experiments were run with $\gamma = 0.9$. For the first set of experiments, we used ϵ-greedy exploration with $\epsilon = 0.4$, decayed linearly to 0 over 10000 episodes. For the constraint defying experiments, we used $\epsilon = 0.4$ and $\upsilon = 0.8$, both decayed linearly to 0 over 10000 episodes.

3. RESULTS

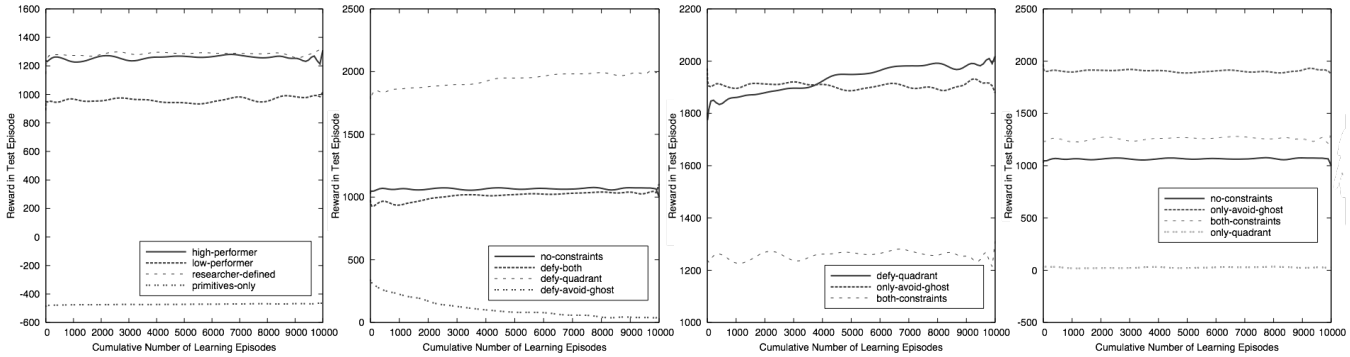
The results of the first experiment for the original size board are presented in Figure 1a (other results are omitted for space). The agent using the HP options and constraints performed significantly better than the one using LP options and constraints (two-sample t-test, $p < 0.001$). Both agents performed significantly better than the primitives only agent (two-sample t-test, $p < 0.001$ for both agents). There was no significant difference between the high performers and the researcher defined options and constraints.

If we allow an HP agent to defy its two constraints, we observe different performance scenarios (Fig. 1b). We can see vastly superior performance from the agent that obeys the avoidGhost constraint and defies the quadrant constraint (a constraint HP players described, not leaving a quadrant until all the food is clear). Thus, it seems to be bad to follow the quadrant constraint all the time.

If we compare the defy-quadrant agent against an agent that obeys only the avoidGhost constraint, we can see both recovery from a bad constraint, and improvement over not using the constraint (Fig. 1c). Mean score increased significantly when using and defying the quadrant constraint over using only the avoidGhost constraint (between 5.08 and 277.23 point difference, two sample t-test, $p < 0.05$); this result is also supported visually by the plot lines in Figure 1c, where a visible and growing gap can be seen between the top two lines after about 4000 episodes.

From the the data presented in Figures 1b and 1c, we see that defying the right constraint can allow an agent to uti-

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*



(a) WDA (b) Defy constraints (c) Partially defy one constraint (d) Explore constraints

Figure 1: Experimental results from WDA study, defiance, partial defiance, and exploration on original sized board.

lize constraints to it's advantage; defying the quadrant constraint while obeying the avoidGhost allowed agent to both recover from a bad constraint, and improve over not using the quadrant constraint at all. We also see that defying the wrong constraint yields very poor performance. Therefore, a method to identify the usefulness of a constraint for the current learning problem is needed when incorporating human task input of this type.

4. APPROACH

To identify the usefulness of a constraint, we compared the impact of each constraint on an agent's ability to learn against an agent that uses both constraints and an agent that uses no constraints; this method was suggested by the defiance results in Figure 1b. The results of this experiment are presented in Figure 1d. This experiment shows after the first episode that the agent using only the avoidGhost constraint performs significantly better than than both the options-only agent and the agent that utilizes both constraints (two-sample t-test, $p < 0.001$).

From this, we generalized an iterative deepening algorithm to evaluate constraints provided by humans based on the rules declared above and two new constraints: one of which should be defied, and one of which should be obeyed. Our approach assumes an episodic MDP with a large or infinite state space in which some states are visited more than once; a way to explore constraints safely; and a finite training period. We assume that we can run at least N trials, where N is sufficiently large so that our sampled reward after each episode will be normally distributed.

An agent can use Algorithm 1 to construct C_D, the constraint to defy; and C_O the constraint to obey. These constraints can then be plugged into ϵ-greedy v-defiant exploration. The time required to run this algorithm is bounded by the number of training episodes in the trial; in practice, however, we see the terminating condition for the algorithm emerge much sooner in the process.

Algorithm 1 Iterative Deepening Explore Constraints

Require: Constraints $\{C_1, C_2\}$, $N \geq ...$, maximum depth M, depth $d \geq 1$
{Return a tuple with the constraint to defy, and the constraint to obey}
if $d \geq M$ **then**
  return $(C_1 \cup C_2, \emptyset)$
else
  $L_1 \leftarrow$ a set of N agents each with C_1.
  $L_2 \leftarrow$ a set of N agents each with C_2.
  $L_{12} \leftarrow$ a set of N agents L_{12} with $C_1 \cup C_2$.
  Run each of L_1, L_2, L_{12} for d full episodes.
  Construct two-sample Student's t-test for samples with unequal variances, to compare $Mean(L_1)$ to $Mean(L_{12})$ and $Mean(L_2)$ to $Mean(L_{12})$.
  $h_1 \leftarrow Mean(L_1) \neq Mean(L_{12})$, if significant.
  $h_2 \leftarrow Mean(L_2) \neq Mean(L_{12})$, if significant.
  if not h_1 **or not** h_2 **then**
    $d \leftarrow d + 1$
    Repeat from start.
  else
    $C_D \leftarrow \{C_1$ if $Mean(L_1) > Mean(L_{12})$ else $\emptyset\} \cup \{C_2$ if $Mean(L_1) > Mean(L_{12})$ else $\emptyset\}$
    $C_O \leftarrow (C_1 \cup C_2) \setminus C_D$
    return (C_D, C_O)
  end if
end if

5. CONCLUSION

We have demonstrated use of an established modeling technique of cognitive engineering to generate options and constraints for a reinforcement learning agent, and shown that relative performance between the agents using the different options and constraints matches that of the human players. We have also demonstrated that selectively defying constraints provided by humans can allow the agent to make use of even a bad constraint, provided the agent designers can test for which constraint to defy. Finally, we present an algorithm that, given two constraints, can construct two additional constraints, one to defy and one to obey designers to test their constraints as mentioned.

Acknowledgments

This work is supported by Office of Naval Research (ONR) N00014-14-1-0003. The results are the work of the authors, do not necessarily reflect the view of ONR.

6. REFERENCES

[1] R. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial intelligence, 112(1):181–211, 1999.

[2] G. Tokadli and K. M. Feigh. Option and constraint generation using work domain analysis. In IEEE International Conference on System, Man, and Cybernetics (IEEE SMC 2014), San Diego, USA, 2014.