Recognizers

A study in learning how to model temporally extended behaviors

Jordan Frank
jordan.frank@cs.mcgill.ca

Reasoning and Learning Lab
McGill University
http://www.cs.mcgill.ca/~jfrank8/

Joint work with Doina Precup

Background and Motivation

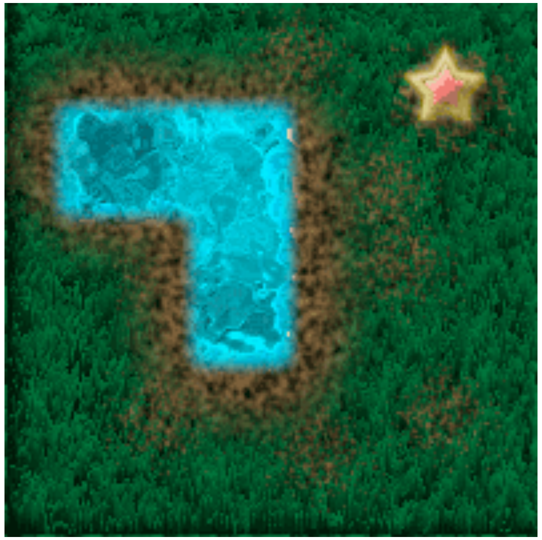
- Want a flexible way to represent hierarchical knowledge. (*Options* [Sutton, Precup & Singh, 1999])
- Want an efficient way to learn about these hierarchies. (*Recognizers* [Precup et al. 2006])
- Concerned with off-policy learning in environments with continuous state and action spaces [Precup, Sutton & Dasgupta 2001].

Terminology

- Option: A tuple $\langle \mathcal{I}, \beta, \pi \rangle$. $\mathcal{I}$ is a set of initiation states, β a termination condition, and π a policy.
- Recognizer: A filter on actions. A recognizer specifies a *class of policies* that we are interested in learning about.
- Off-policy learning: We are interested in learning about a *target policy* π by observing an agent whose behavior is governed by a different (possibly unknown) policy b.

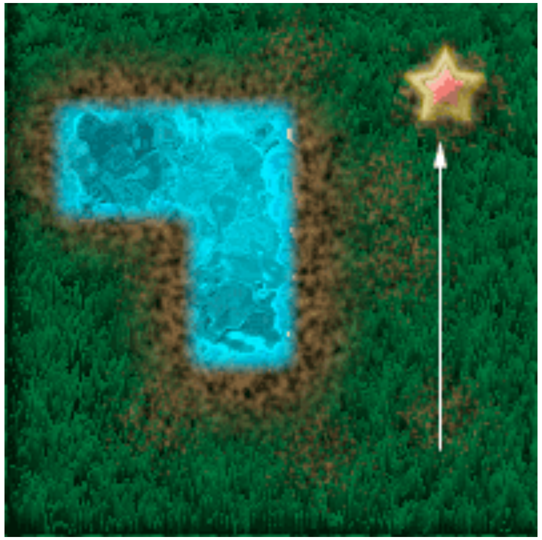
Example Problem

- PuddleWorld [RL-Glue]
  - **Continuous** state space
  - **Continuous** action space
- Goal is to do *off-policy learning*. Behavior policy is *unknown*.



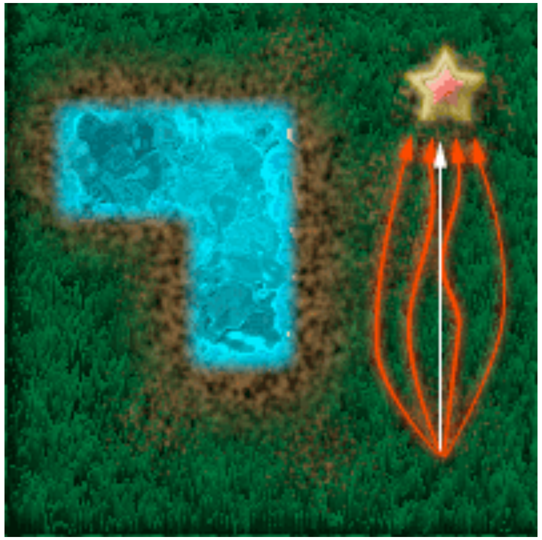
Example Problem

- PuddleWorld [RL-Glue]
  - **Continuous** state space
  - **Continuous** action space
- Goal is to do *off-policy learning*. Behavior policy is *unknown*.



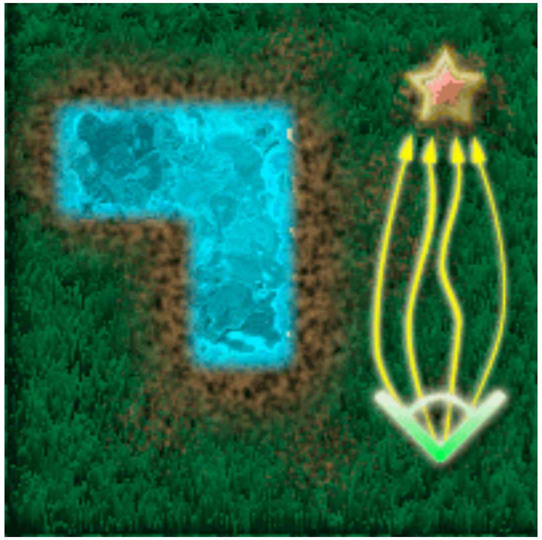
Example Problem

- PuddleWorld [RL-Glue]
  - **Continuous** state space
  - **Continuous** action space
- Goal is to do *off-policy learning*. Behavior policy is *unknown*.



Example Problem

- PuddleWorld [RL-Glue]
  - **Continuous** state space
  - **Continuous** action space
- Goal is to do *off-policy learning*. Behavior policy is *unknown*.



Recognizers: Formally

- MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$. At time step t, an agent receives a *state* $s_t \in \mathcal{S}$ and chooses an *action* $a_t \in \mathcal{A}$.
- Fixed (unknown) *behavior policy* $b : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$, used to generate actions.
- Recognizer is a function $c : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$, where $c(s, a)$ indicates to what extent the recognizer allows action a in state s.
- *Target policy* π generated by b and c

$$\pi(s, a) = \frac{b(s, a)c(s, a)}{\sum_x b(s, x)c(s, x)} = \frac{b(s, a)c(s, a)}{\mu(s)},$$

where $\mu(s)$ is the *recognition probability* at s.

Importance Sampling

- Based on the following observation:

$$E_\pi\{x\} = \int_x x\pi(x)dx = \int_x x\frac{\pi(x)}{b(x)}b(x)dx = E_b\left\{x\frac{\pi(x)}{b(x)}\right\}$$

- We are trying to learn about a target policy π using samples drawn from a behavior policy b, and so we just need to calculate the appropriate weights.

- Weights (also called *corrections*) given by

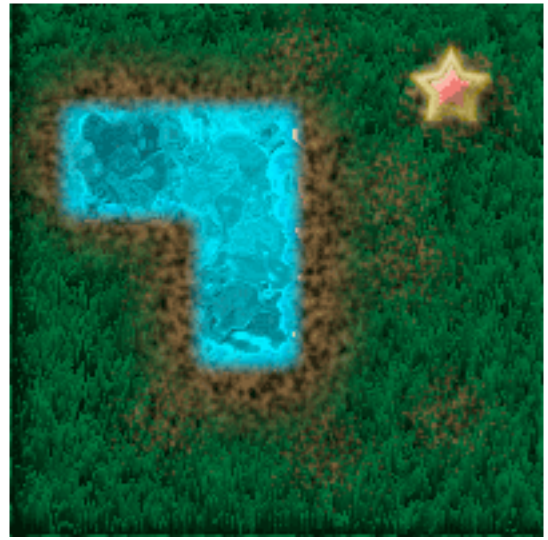
$$\rho(s,a) = \frac{\pi(s,a)}{b(s,a)} = \frac{c(s,a)}{\mu(s)}$$

- Full details of the algorithm given in Precup et al. (2006).

Importance Sampling Correction

- $\mu(s)$ depends on b.
- If b is *unknown*, we can use a *maximum likelihood estimate* $\hat{\mu} : \mathcal{S} \rightarrow [0, 1]$.
- For *linear function approximation*, we can use *logistic regression* with the same set of features in order to estimate μ.

Experiment 1: Puddle World [RL-Glue]

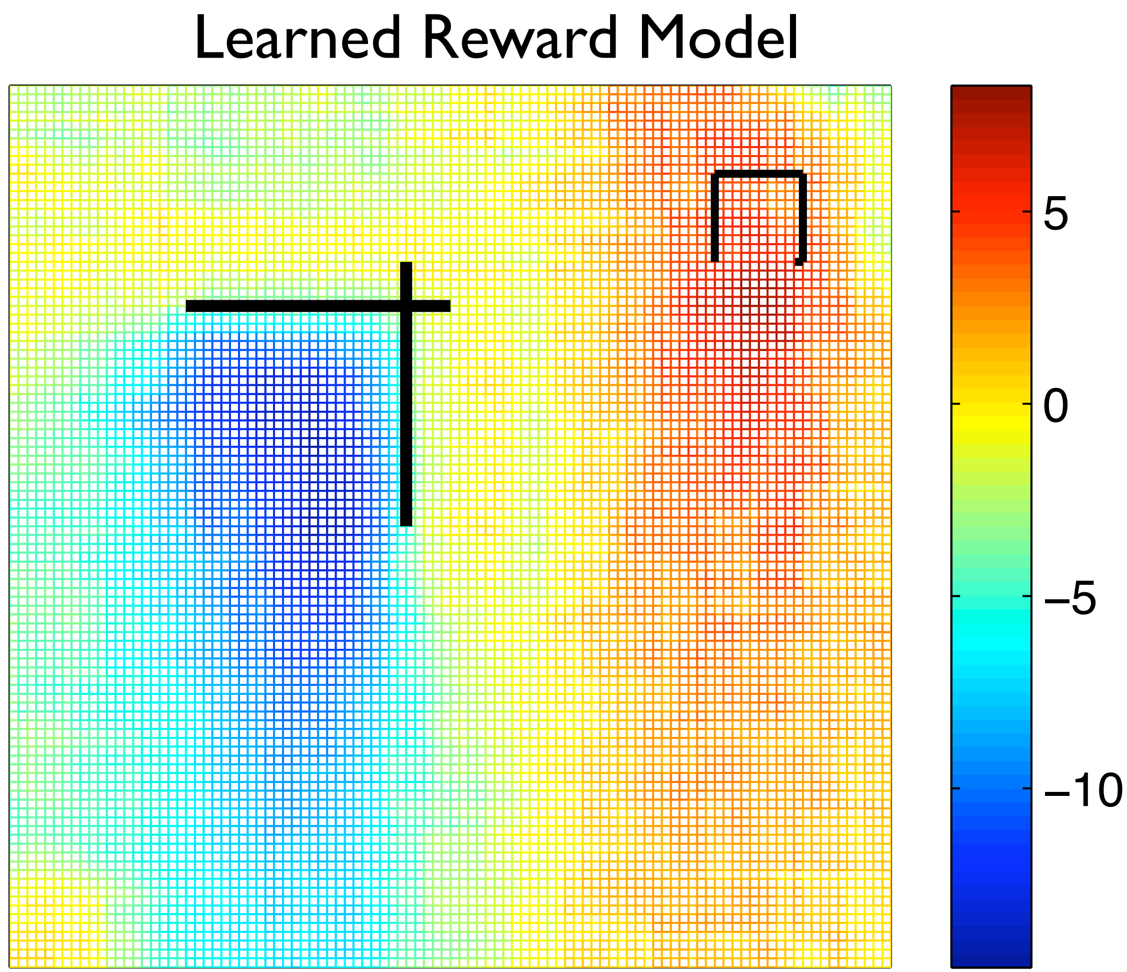


- Continuous state space, continuous actions. Movement is noisy.
- Positive reward for reaching goal (10), negative reward for entering puddle (-10 at middle).
- Start state chosen randomly in small square in lower left corner. Reaching goal moves agent back to start state

Experiment 1: Setup

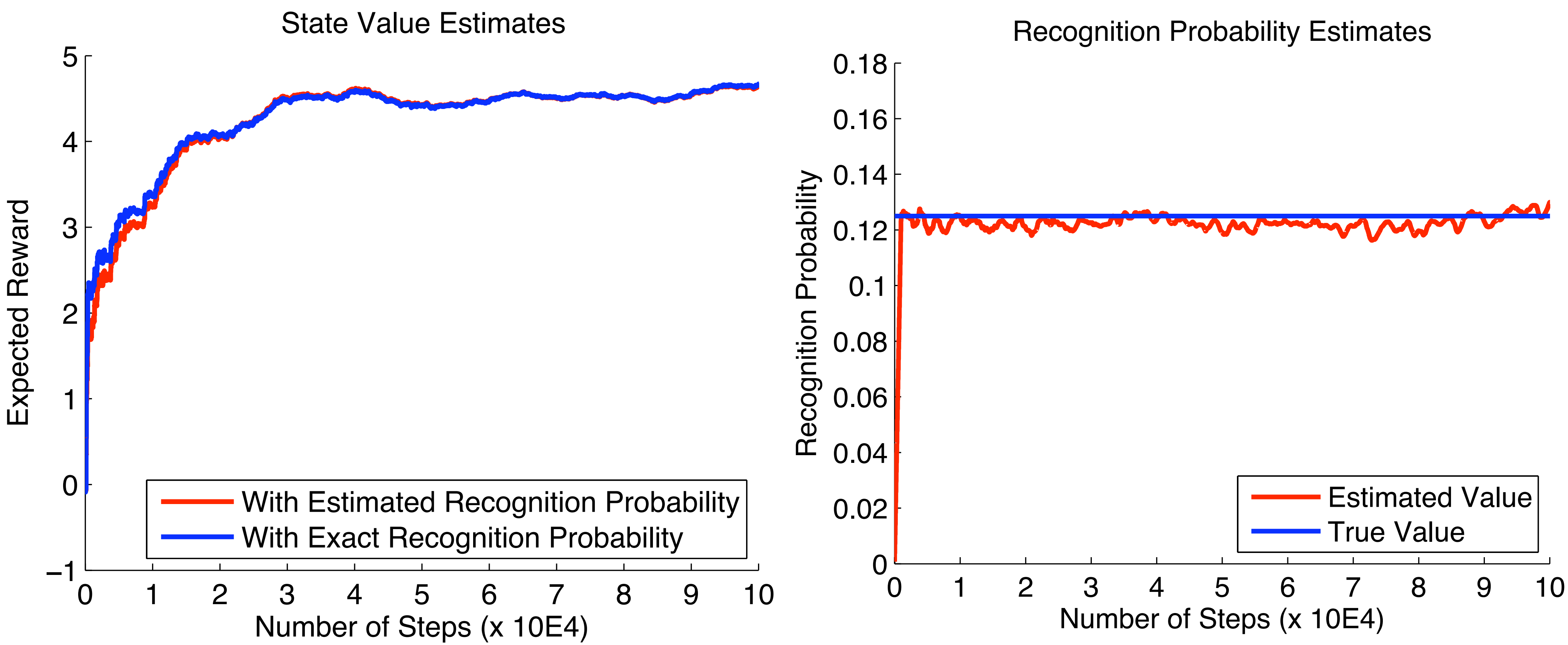
- Standard tile coding *function approximation* for state space.
- Behavior policy picks actions uniformly randomly, target policy is to pick actions that lead *directly* towards the goal state.
- Binary recognizer, recognizes actions in a 45° cone facing directly towards the goal state. Recognizer episode can be initiated everywhere, and terminates when either goal state or puddle are entered.

Experiment 1: Results



- This matches our intuition that moving directly towards the goal is good unless you are below and to the left of the puddle.

Experiment 1: Results



- We observe that the recognition probability estimate converges to the correct value, and estimating this value as we do our learning does not bias our state value estimates.

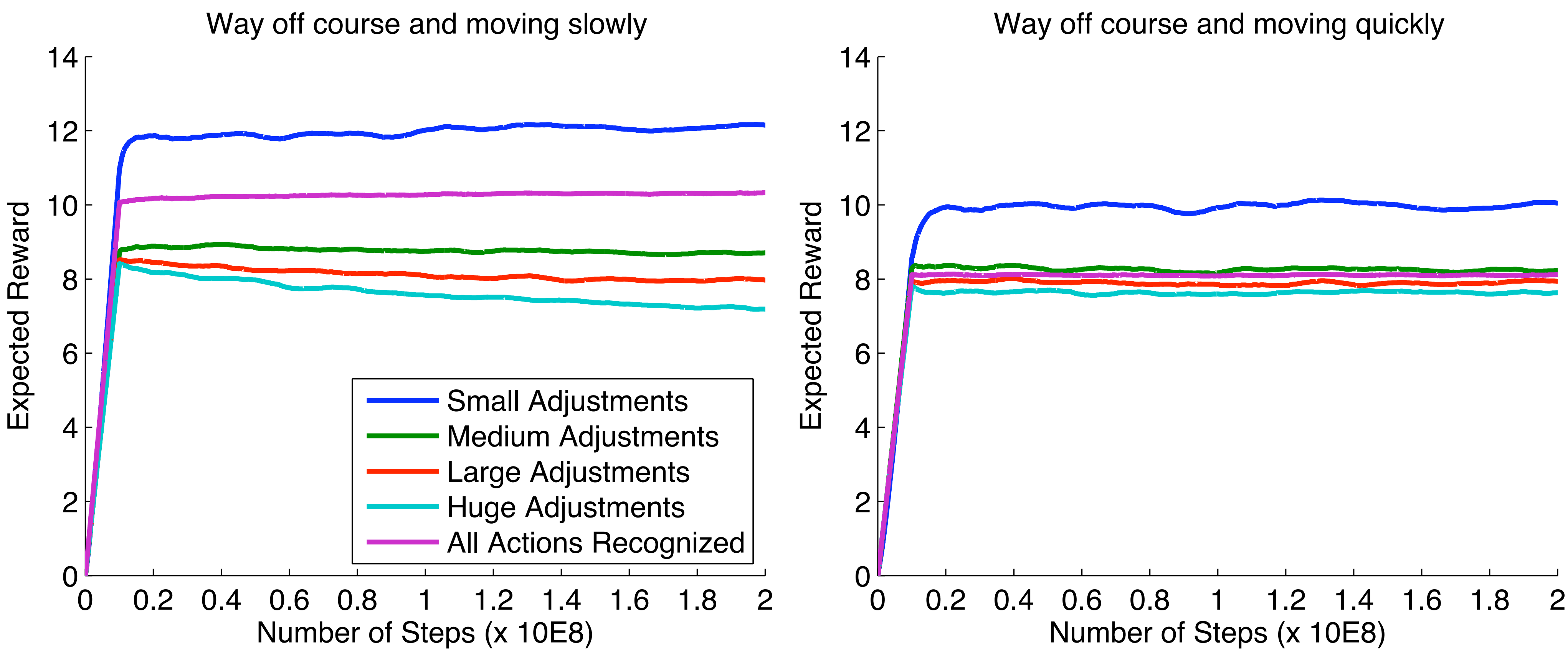
Experiment 2: Ship Steering [RL-Glue]

- Stochastic environment. 3D Continuous state space, 2D continuous actions (throttle and rudder angle).
- Goal is to keep a ship on a desired heading with a high velocity.

Experiment 2: Setup

- Goal is to demonstrate that we can learn multiple recognizers from one stream of experience.
- Behavior policy picks a rudder orientation randomly to bring ship towards desired heading.
- 4 recognizers recognize different ranges of motion, from small, smooth adjustments to the rudder, to huge, sharp adjustments.

Experiment 2: Results



- We can see that policies that make smaller rudder adjustments outperform those that make large adjustments.

Conclusion and Future work

- Recognizers are useful for learning about options when we cannot control, or do not know the behavior policy.
- Convergence has been shown for state aggregation, still need to work on proofs for function approximation, but empirical results are promising.
- More experiments.

Questions?

- RL-Glue, University of Alberta, *http://rlai.cs.ualberta.ca/RLBB/top.html*

- Precup, D., Sutton, R.S., and Dasgupta, S. (2001). Off-policy temporal-difference learning with function approximation. In *Proc. 18th International Conf. on Machine Learning*, pages 417-424.
- Precup, D., Sutton, R.S., Paduraru, C., Koop, A., and Singh, S. (2006). Off-policy learning with recognizers. In *Advances in Neural Information Processing Systems 18 (NIPS*05)*.
- Sutton, R.S., Precup, D., and Singh, S.P. (1999). Between MDPs and semi-MDPS: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2): 181-211.