Value Function Approximation with Hierarchical Clustering in Soar-RL

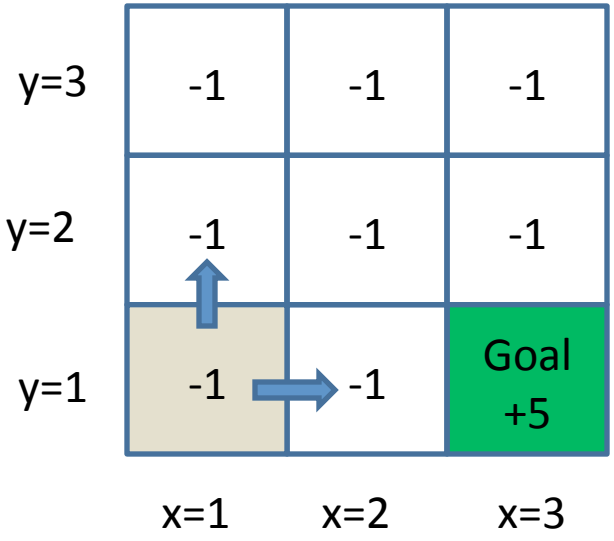
Yongjia Wang

University of Michigan

Soar-RL Value Function

- Soar-RL relies on learning the Q value function
  - Q Value Function, Q(state, action)

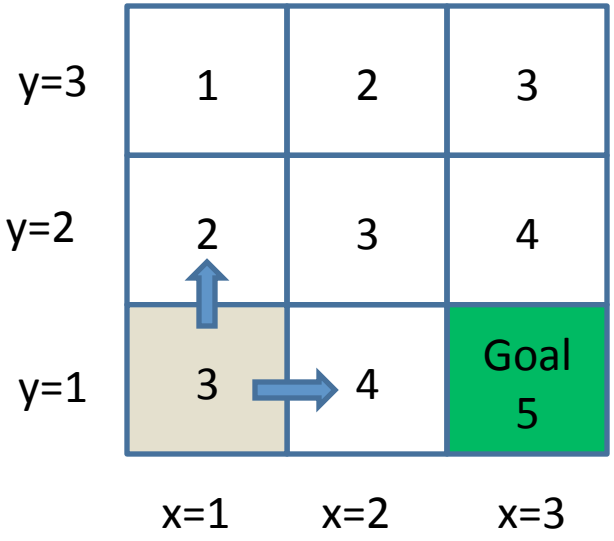
(x=1, y=1), (move-up) --> Q=1
(x=1, y=1), (move-right) --> Q=3
(x=1, y=1), (move-down) --> Q=2
(x=1, y=1), (move-left) --> Q=2



Soar-RL Value Function

- Soar-RL relies on learning the Q value function
  - Q Value Function, Q(state, action)
  - Value(state) = max(Q(state, action_i))+r

(x=1, y=1), (move-up) --> Q=1
(x=1, y=1), (move-right) --> Q=3
(x=1, y=1), (move-left) --> Q=2
(x=1, y=1), (move-down) --> Q=2

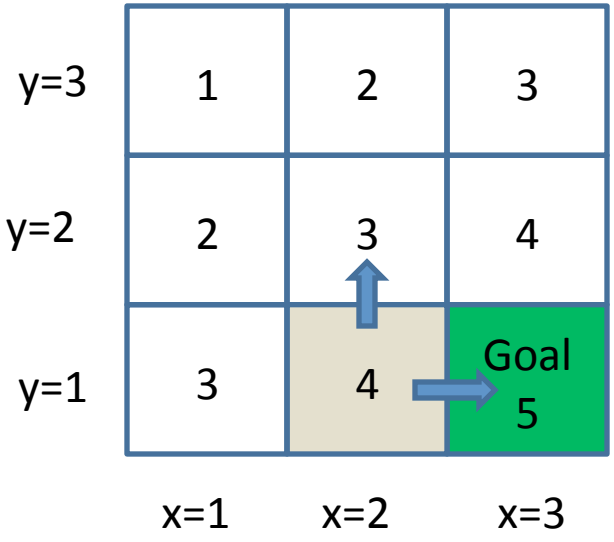


Soar-RL Value Function

- Soar-RL relies on learning the Q value function
  - Q Value Function, Q(state, action)
  - Value(state) = max(Q(state, action_i))

(x=1, y=1), (move-up) --> Q=1
(x=1, y=1), (move-right) --> Q=3
(x=1, y=1), (move-left) --> Q=2
(x=1, y=1), (move-down) --> Q=2

(x=2, y=1), (move-up) --> Q=2
(x=2, y=1), (move-down) --> Q=3
(x=2, y=1), (move-left) --> Q=2
(x=2, y=1), (move-right) --> Q=4



Soar-RL Value Function

- Soar-RL relies on learning the Q value function
  - Q Value Function, Q(state, action)
  - Value(state) = max(Q(state, action_i))
  - What about high dimensional continuous space?

    (x=1, y=1, z=1, … …)
    (x=1.1, y=1, z=1, … …)
    (x=1.2, y=1, z=1, … …)
    … …

  - Need Value Function Approximation

Motivations

- Improve Value Function Approximation in Soar-RL
  - High dimension continuous space
- Evaluate with challenging tasks
  - Characterize the tasks
  - Compare with baselines

Outline

- Background
- Evaluation Task
- Implementation
- Results
- Nuggets & Coal

Value Function Approximation -Parametric Form

Simple linear model:

Q(x,y,up) = $\mathbf{w_1}x + \mathbf{w_2}y + \mathbf{w_3}$

Q(x,y,down) = $\mathbf{w_1}x + \mathbf{w_2}y + \mathbf{w_4}$

Q(x,y,left) = $\mathbf{w_1}x + \mathbf{w_2}y + \mathbf{w_5}$

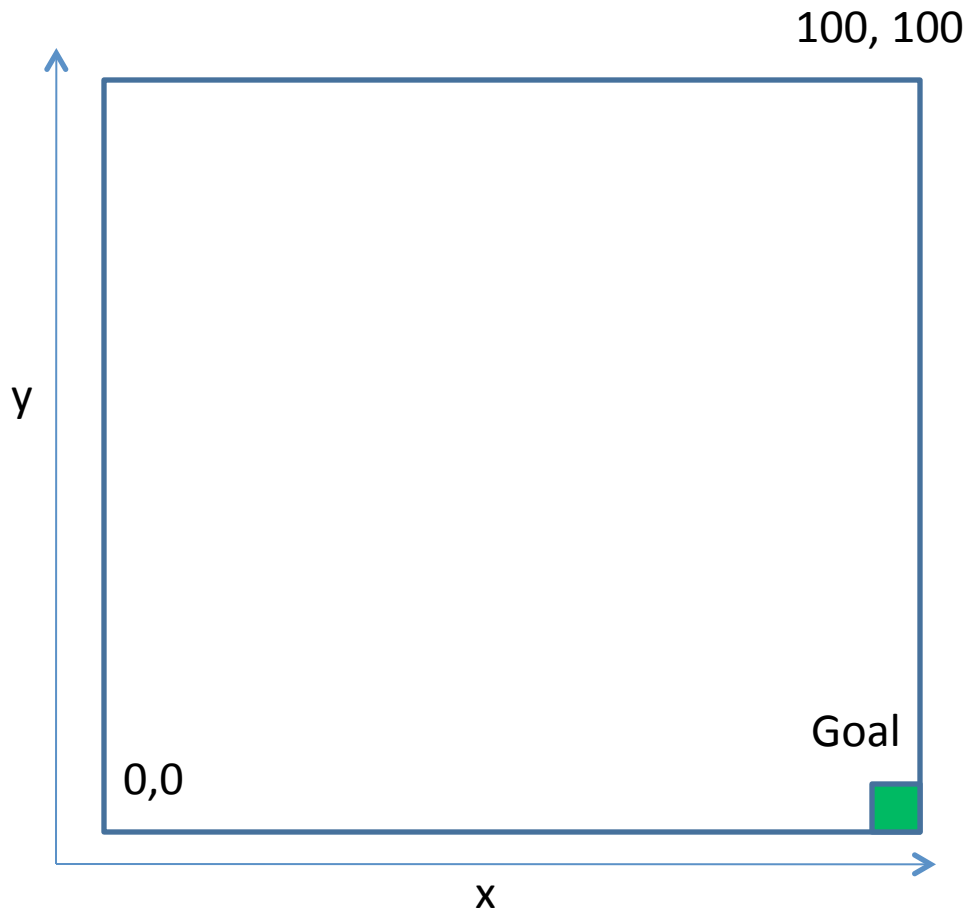
Q(x,y,right) = $\mathbf{w_1}x + \mathbf{w_2}y + \mathbf{w_6}$

Solution:

$\mathbf{w_1 > 0, w_2 < 0}$

$\mathbf{w_3 < w_4}$

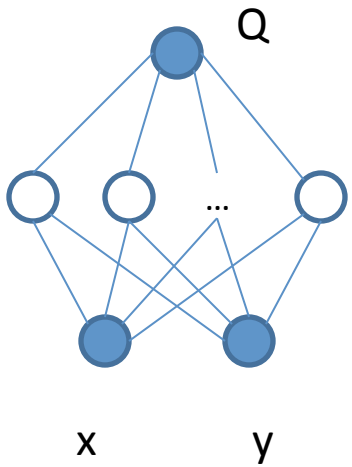
$\mathbf{w_5 > w_6}$

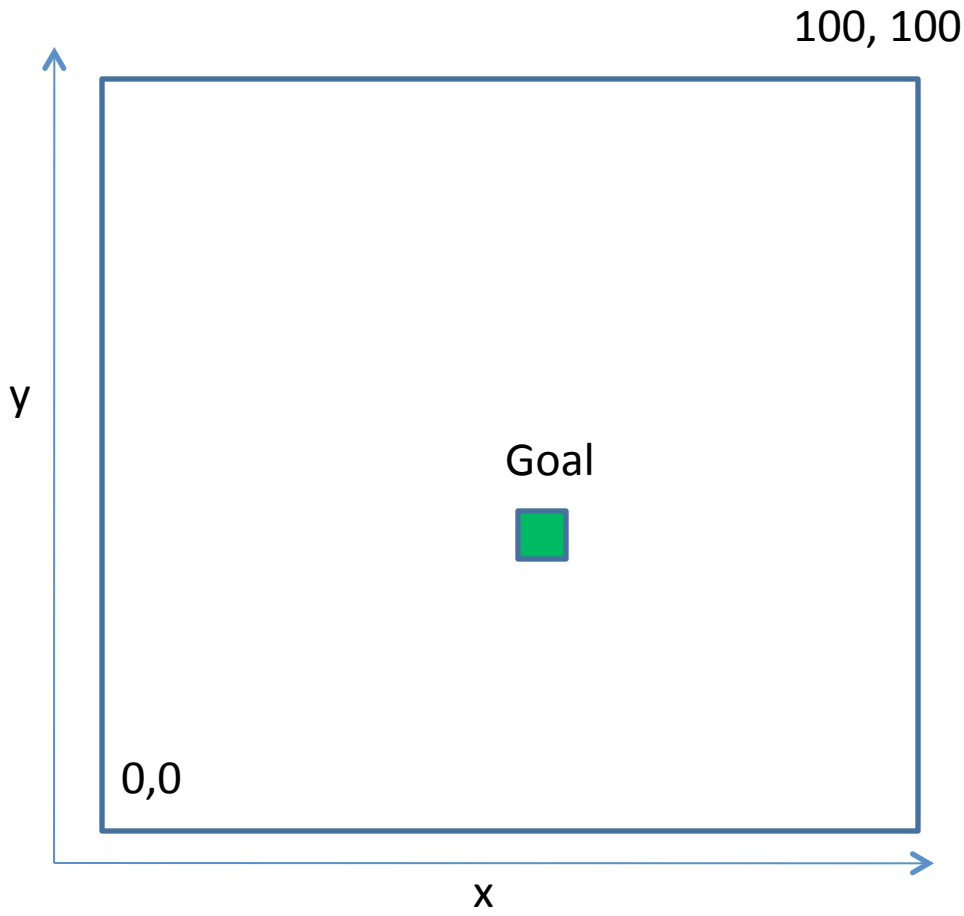


Value Function Approximation -Parametric Form

Simple linear model won't work.
Need nonlinear basis function.

Neural network:

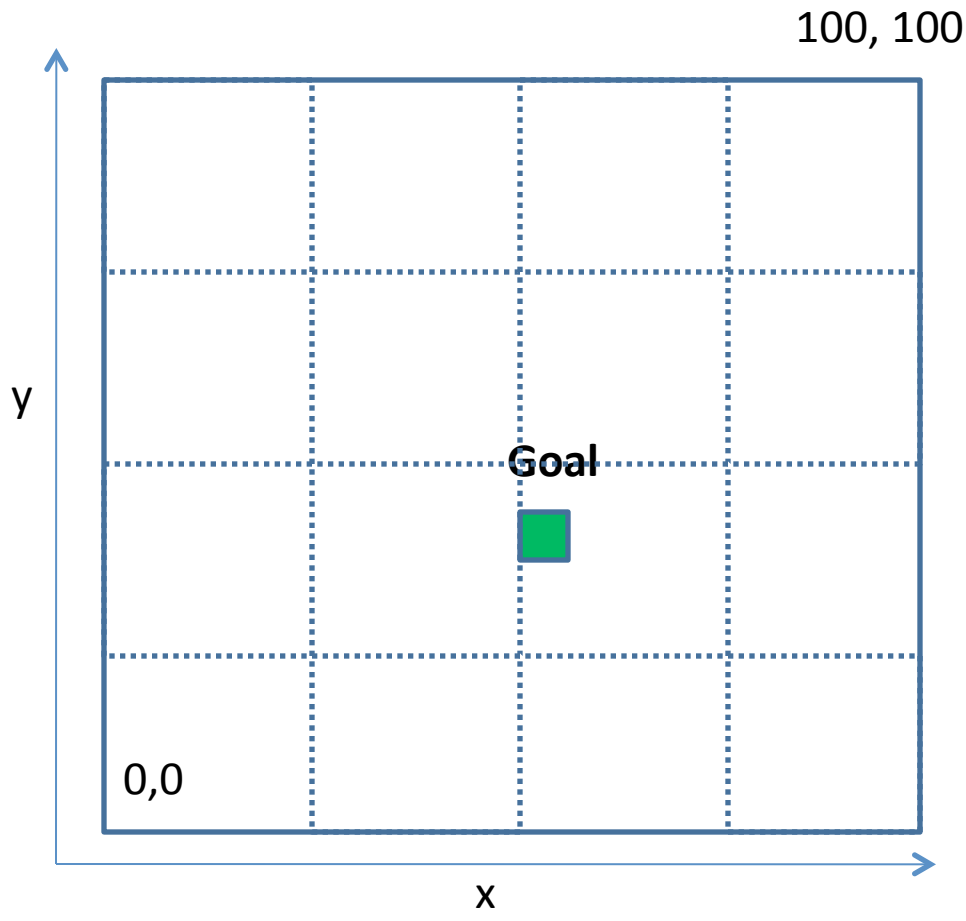




Global Approximators: updates in parameters during learning affects the entire state space

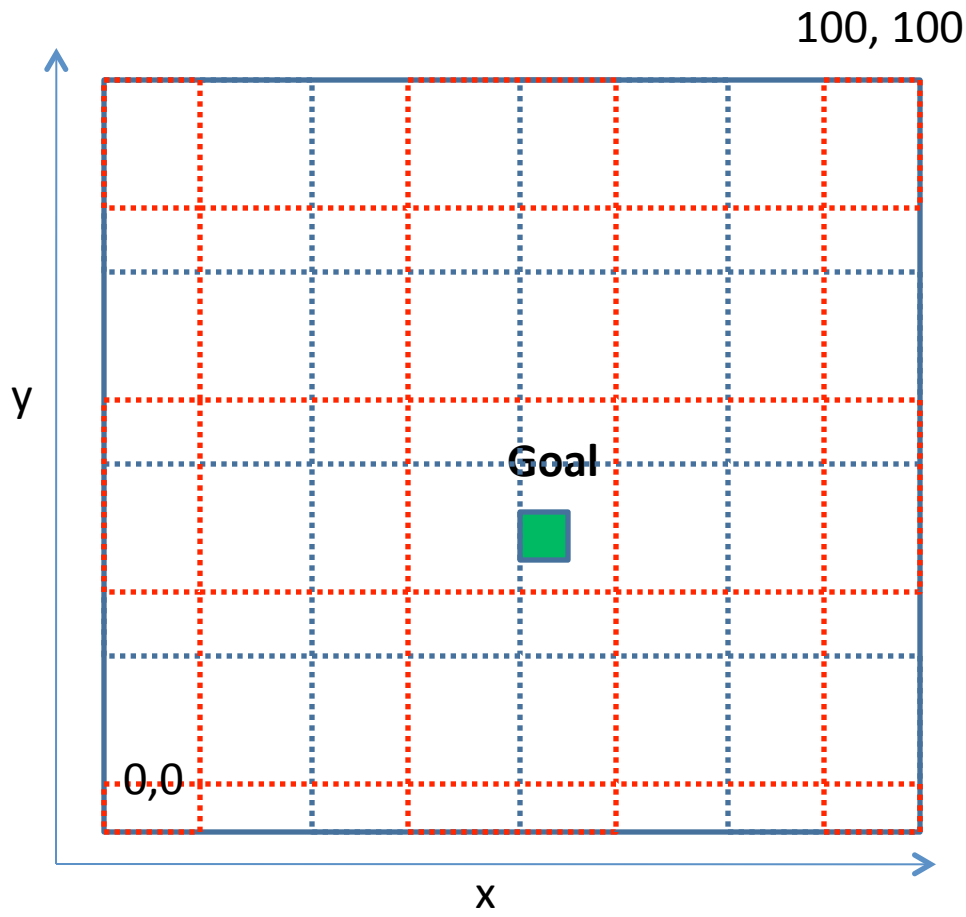
Value Function Approximation -Nonparametric methods

Tile coding



Value Function Approximation -Nonparametric methods

Coarse coding



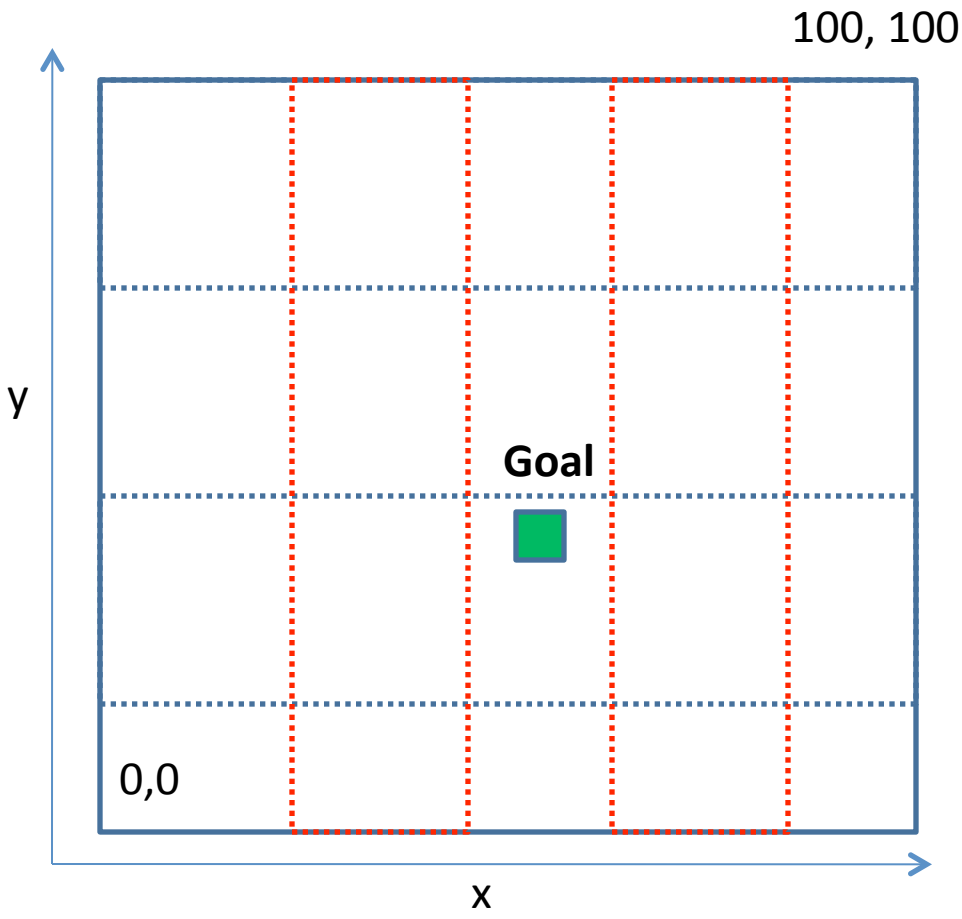
Value Function Approximation -Nonparametric methods

Soar-RL supports arbitrary coarse coding by using "coarse features" in RL rules

RL rule_1: ($x_0 < x < x_1$) -> $Q=q_1$
RL rule_2: ($y_0 < y < y_1$) -> $Q=q_2$
… …
RL rule_n

$Q(x_0 < x < x_1, y_0 < y < y_1) = q_1+q_2$

How to decide the boundaries?
How to encode rules for high dimensional space?

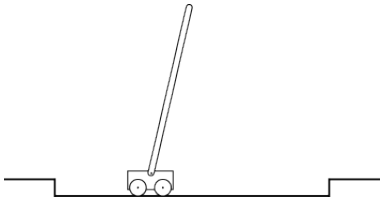


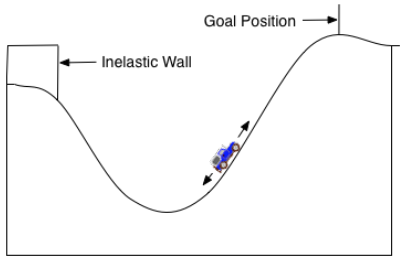
Motivations

- Improve Value Function Approximation in Soar-RL
  - Manually design coarse features can be challenging, even infeasible
  - Learn coarse features automatically
  - Compatible with existing VFA scheme in Soar-RL
- Evaluate with challenging tasks
  - Characterize the tasks
  - Compare with baselines

Traditional RL

- Dynamic control problem
  - Pole balancing
  - Mountain car
  - Robotic arm
  - Helicopter





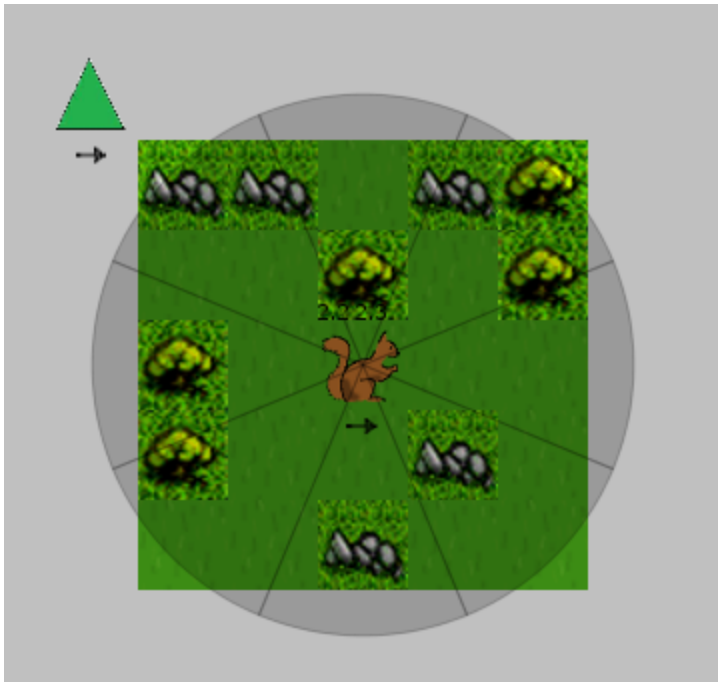
Object Oriented RL

- Object Oriented MDP (OOMDP) framework
  - *Diuk* et al. 2008
  - Representation and learning of action models, not value function approximation
  - Limited object diversity

Motivations

- Improve Value Function Approximation in Soar-RL
  - Manually design coarse features can be challenging
  - Learn coarse features by hierarchical clustering
  - Compatible with existing VFA scheme in Soar-RL
- Evaluate with challenging tasks (Object oriented environments)
  - State representation consists of objects
  - Objects are functionally and perceptually diverse, represented by feature vectors
  - Complex interactions among objects

Evaluation Environment



Distance: 3
Obstacles: bush at 1, rock at 2

Prey:

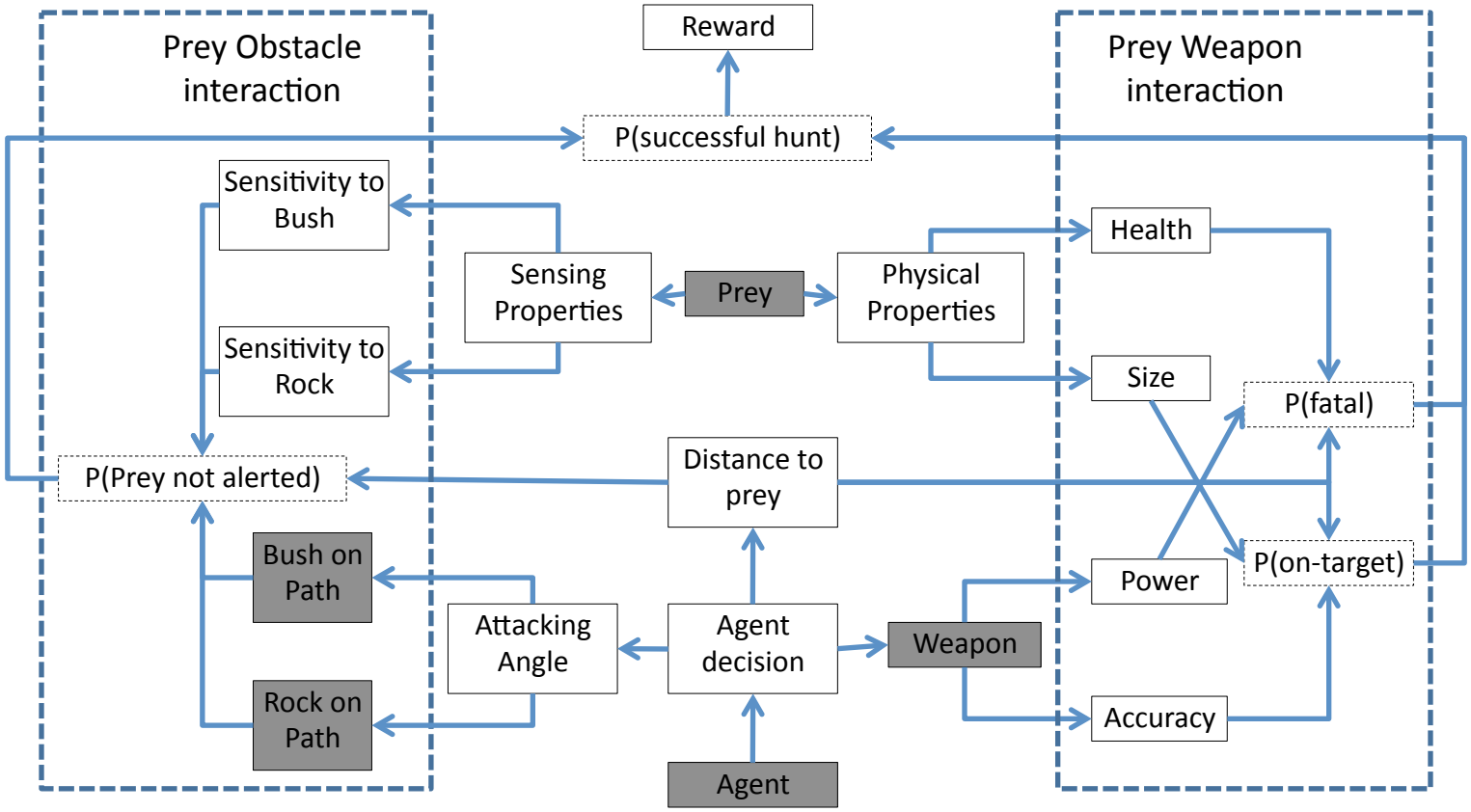
size	health	bush-sensitivity	rock-sensitivity
1	2	2	4

Weapon:

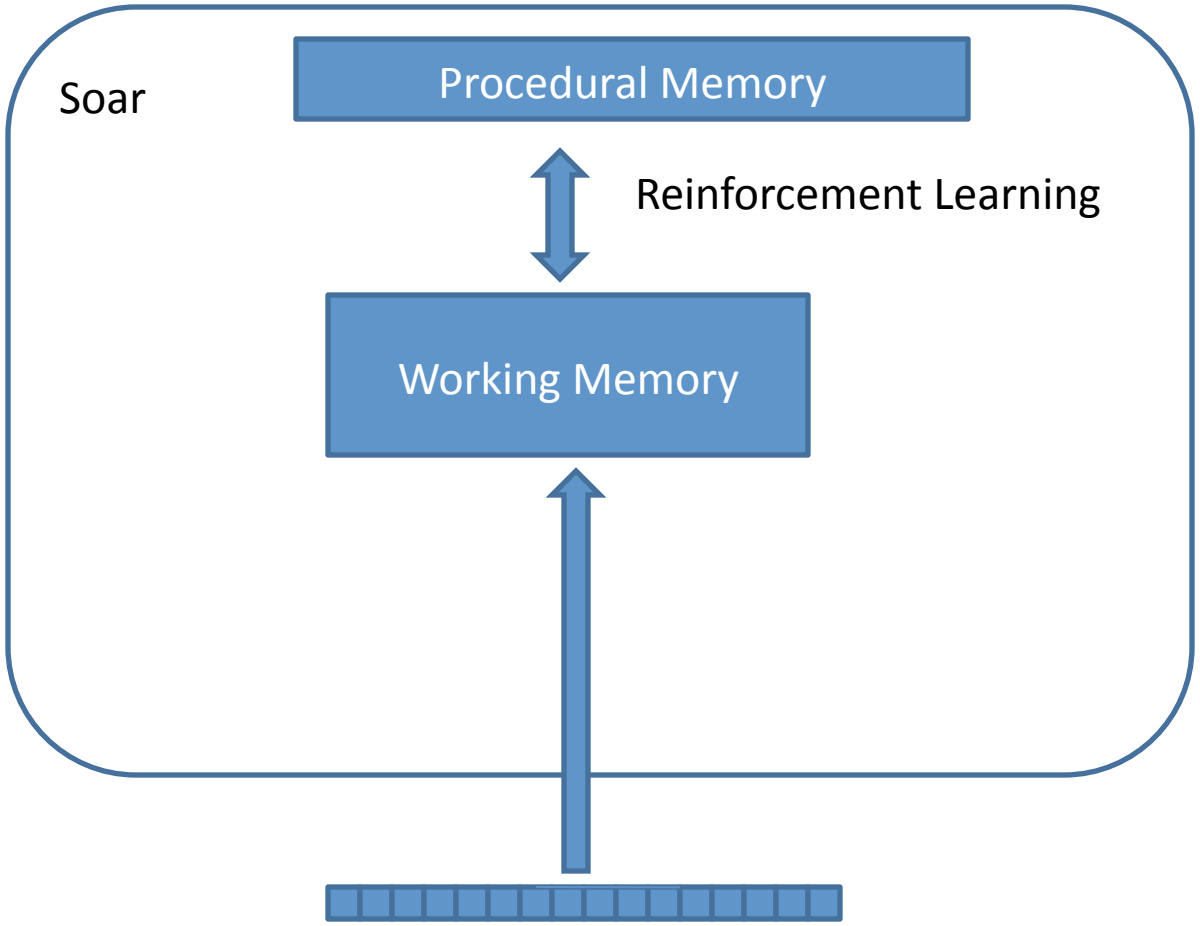
accuracy	power
5	2

1. Not alert the prey
2. Shot on target
3. Fatally wound the prey
4. Reward

Environment Model

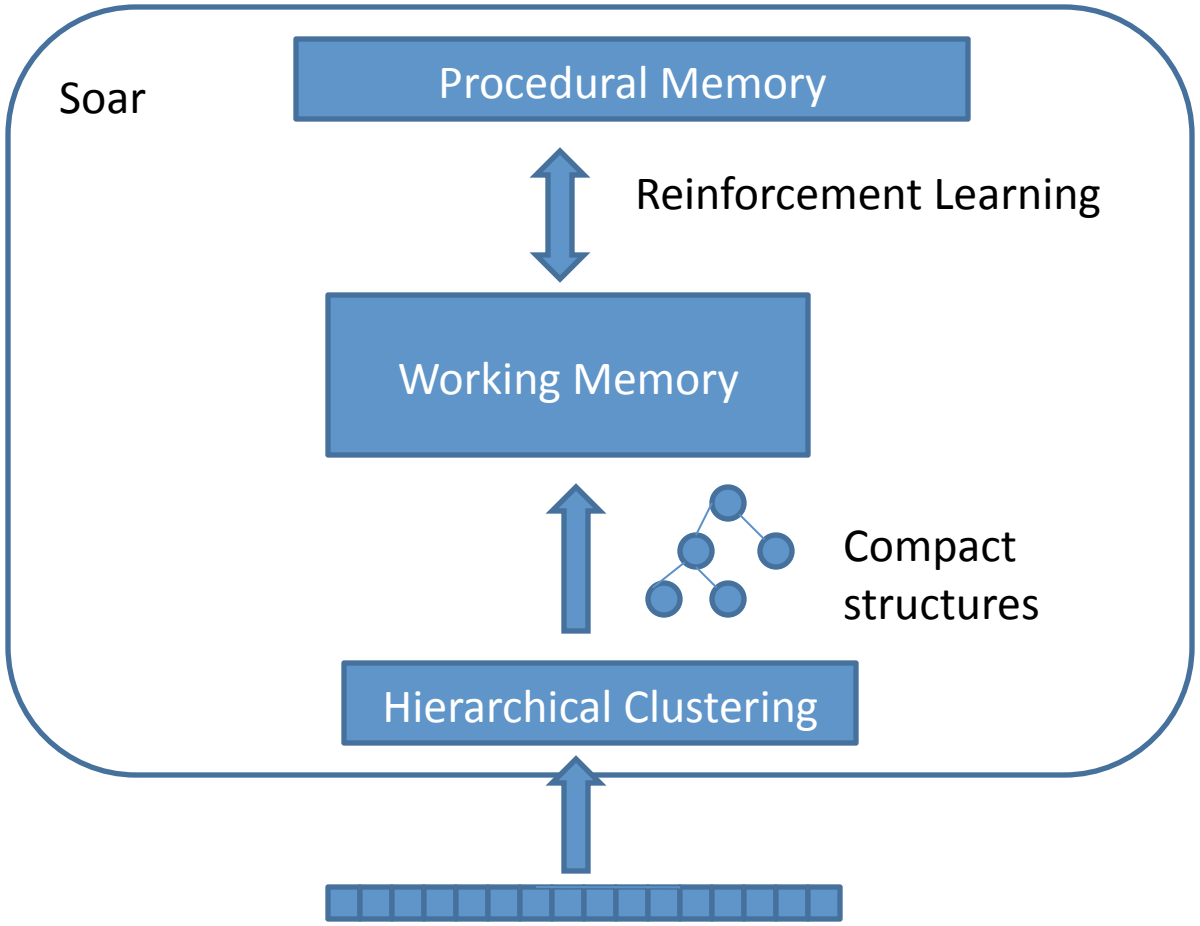


Integrate Hierarchical Clustering with Soar-RL



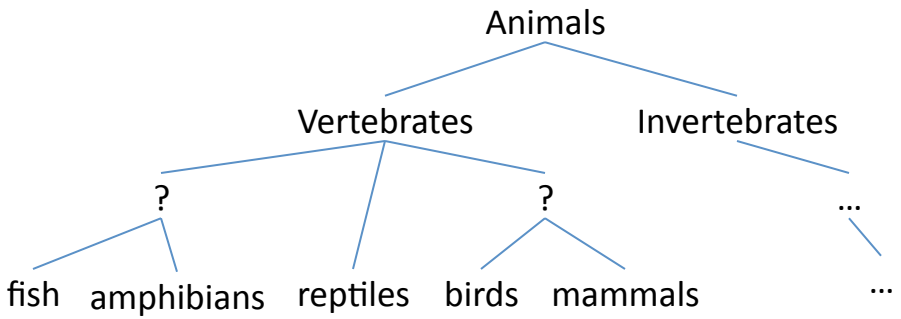
High dimension, continuous state space

Integrate Hierarchical Clustering with Soar-RL



Integrate Category Learning with RL

- COBWEB (D. Fisher 1987)
  - Incremental learning
    - process one instance a time
  - Hierarchical clustering
    - Take numeric feature vector as input
    - Automatically create category symbols
    - Hierarchical structure

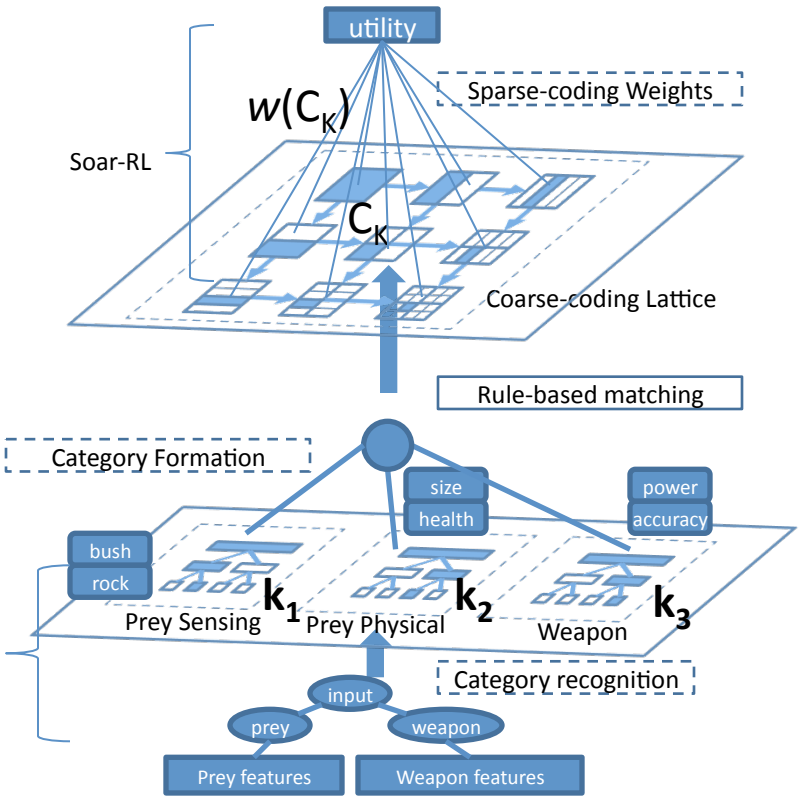


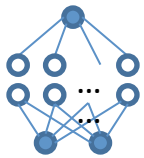
Details of the Algorithm

Procedural Memory

Working Memory

COBWEB





Vector K = (k_1, k_2, k_3), represents the activated clusters from each of the three hierarchies.

A cell C_K, represents the rule matching the clusters K, and $a(C_K)$ is the activation:

$$a(C_K) = \begin{cases} 1 \; if \; matches \\ 0 \; if \; not \; matches \end{cases}$$

Connection weight $w(C_K)$ is the Q value associated with the RL rule. The predicted value:

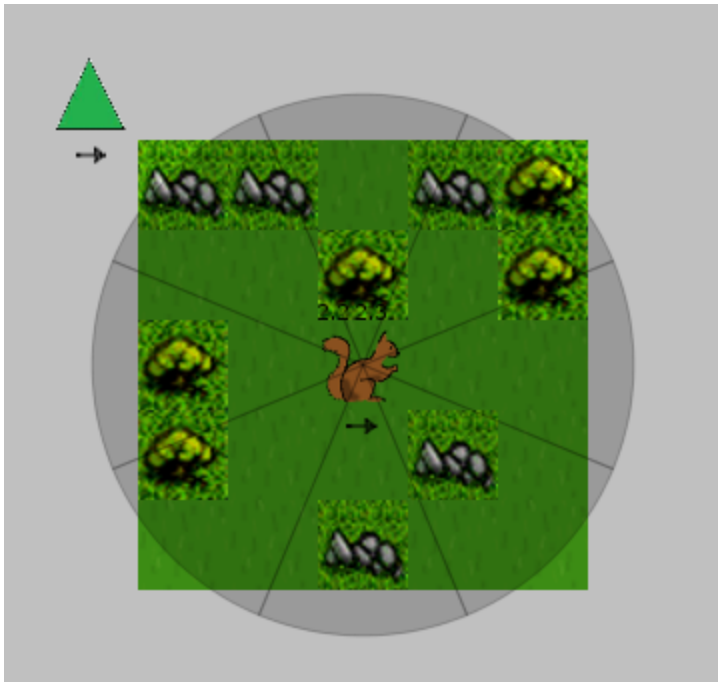
$$y = \sum_{C_K} w(C_K) a(C_K)$$

Update in Soar-RL is calculated in the same way as in stochastic gradient descent methods

$$\Delta w(C_K) = \frac{\alpha}{\sum_{C_K} a(C_K)} (t - y) a(C_K)$$

Like two-hidden-layer network, except for competitive learning and sparse activity in the "hidden layer".

Evaluation Environment



Distance: 3
Obstacles: bush at 1, rock at 2

Prey:

size	health	bush-sensitivity	rock-sensitivity
1	2	2	4

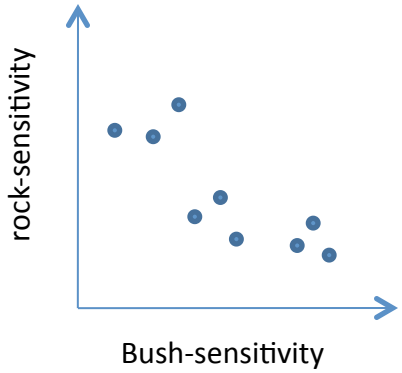
Weapon:

accuracy	power
5	2

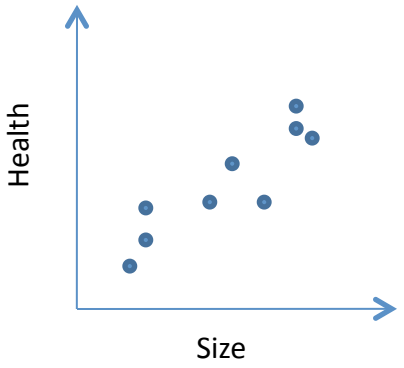
1. Not alert the prey
2. Shot on target
3. Fatally wound the prey
4. Reward

Data Used for Evaluation

Prey

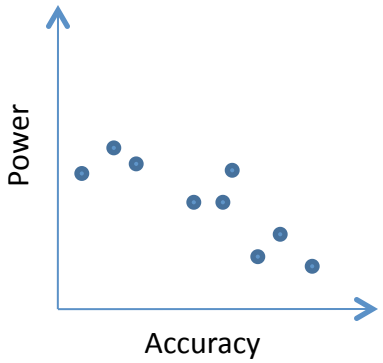


Prey adapted to specific terrain



Health and size are positively correlated

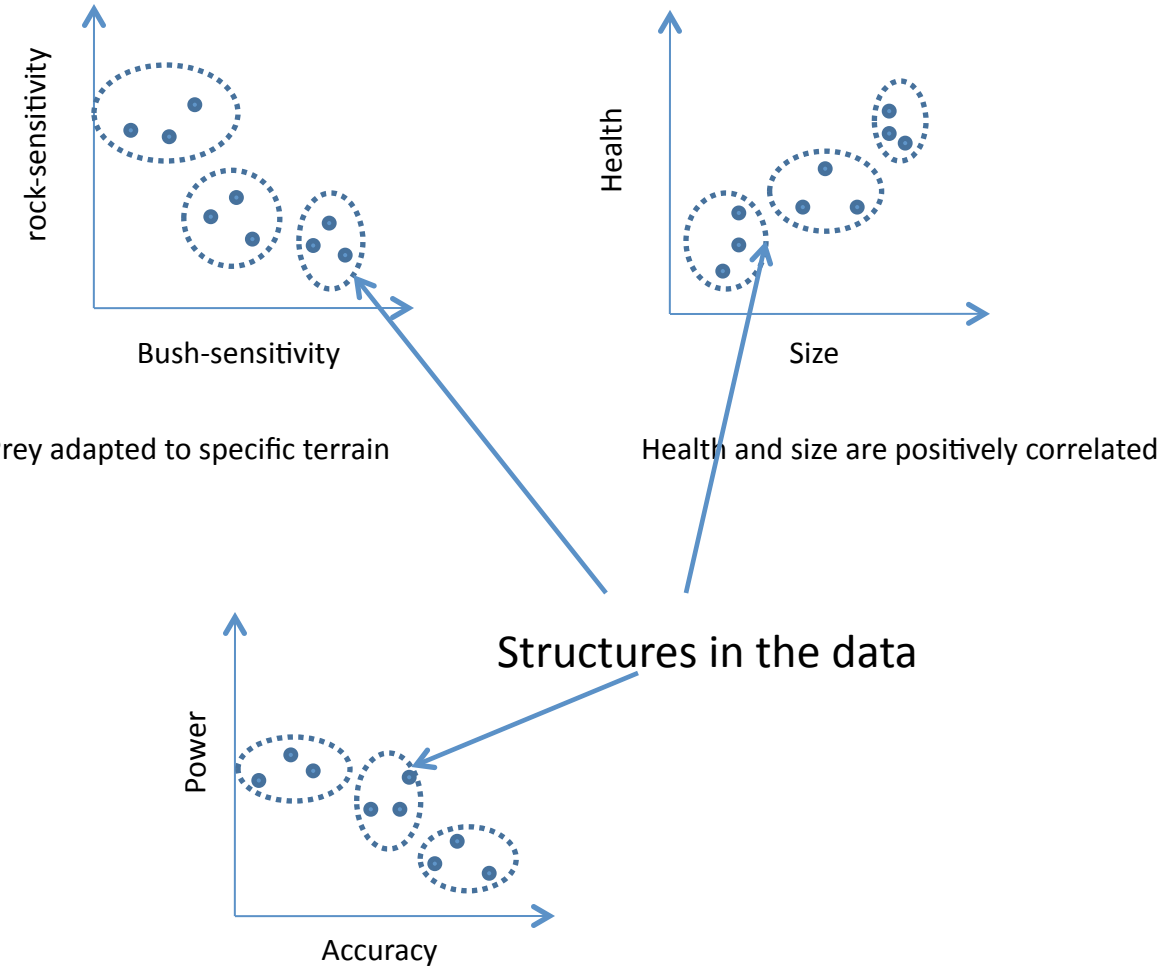
Weapon



Power and accuracy are negatively correlated

Data Used for Evaluation

Prey

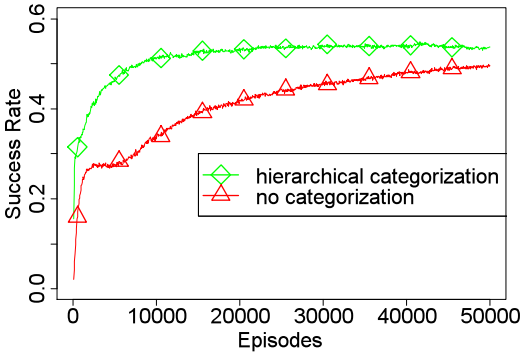


Weapon

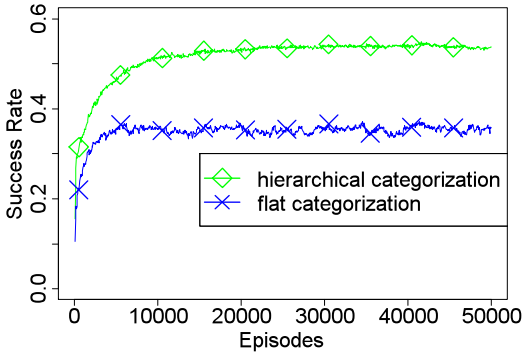
Evaluation

- Target system (Hierarchical clustering + RL)
  - Hierarchical Categorization
  - Using specific functional features
    - prey sensing, prey physical, weapon
- Baseline 1
  - No categorization
  - No generalization, slow learning
- Baseline 2
  - Flat categorization
  - Can be over-general, fail to discriminate more specific cases
- Baseline 3
  - A single hierarchy for prey including all perceptual features
  - Hierarchy with lower quality: some perceptual features are irrelevant to the task (such as color)

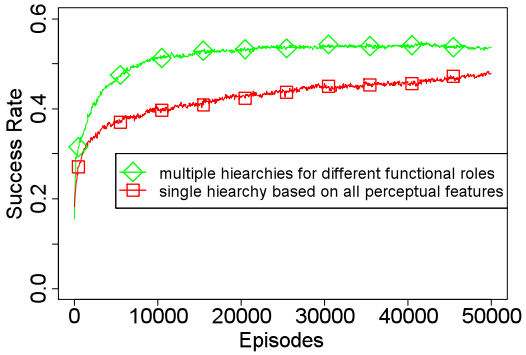
Results



Against baseline1
No generalization



Against baseline2
Over-generalization



Against baseline3
Hierarchy of lower quality

	Categorization	Hierarchical	Functional Hierarchies
Target	Yes	Yes	Yes
Baseline1	No	NA	Yes
Baseline2	Yes	No	Yes
Baseline3	Yes	Yes	No

Nuggets & Coal

- Nuggets
  - Integrated hierarchical category learning and Soar-RL for object-oriented environments with complex features
  - Provide means to take advantage of prior knowledge about functional features
- Coal
  - Single point evaluation
  - Limited complexity in object diversity
  - Only one way flow of information: from categorization to VFA