Reward Shaping

Yun Hua

2019/10/8

Outline

- Introduction to Reward Shaping
- Proof of Policy Invariant
- Further Works
  - 1. Representation of Shaping Function
  - 2. Learning the reward shaping(Choosing a potential function)
  - 3. Other Application(Policy Transfer)
  - 4. Others

Reinforcement Learning

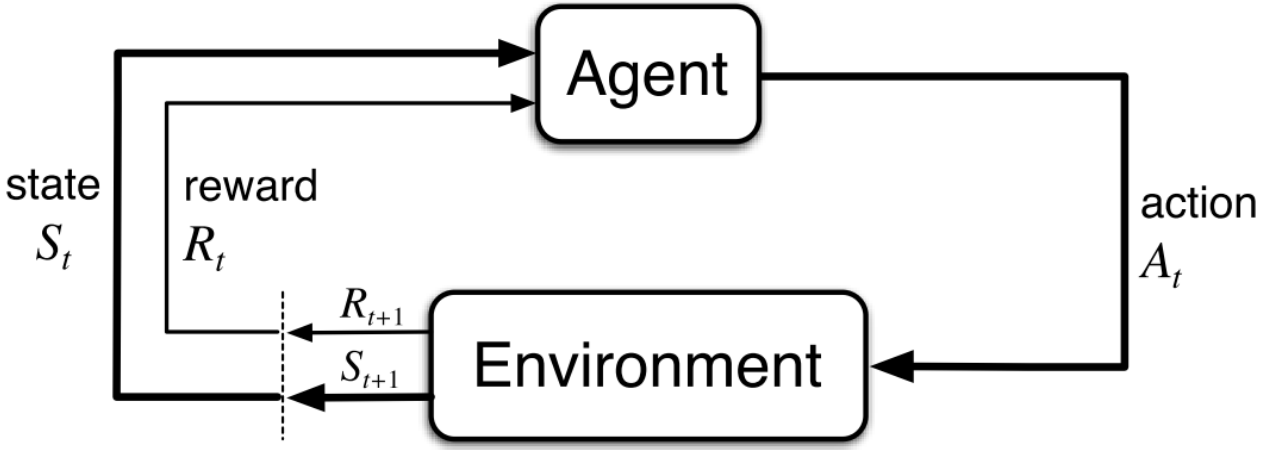
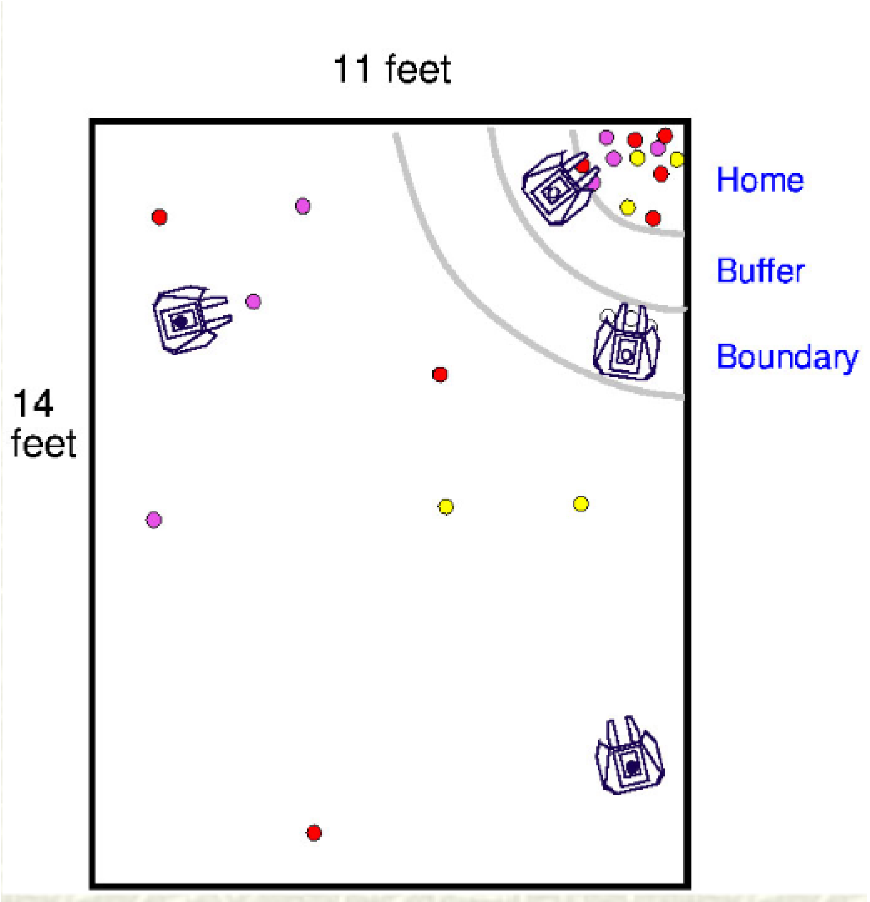


Figure 3.1: The agent–environment interaction in a Markov decision process.

Example: Foraging Robots. Why we need Reward Shaping



The robots' objective is to collectively find pucks and bring them home.

If a reward is given only when a robot drops a puck at home, learning will be extremely difficult. The **delay between the action and the reward is large.**

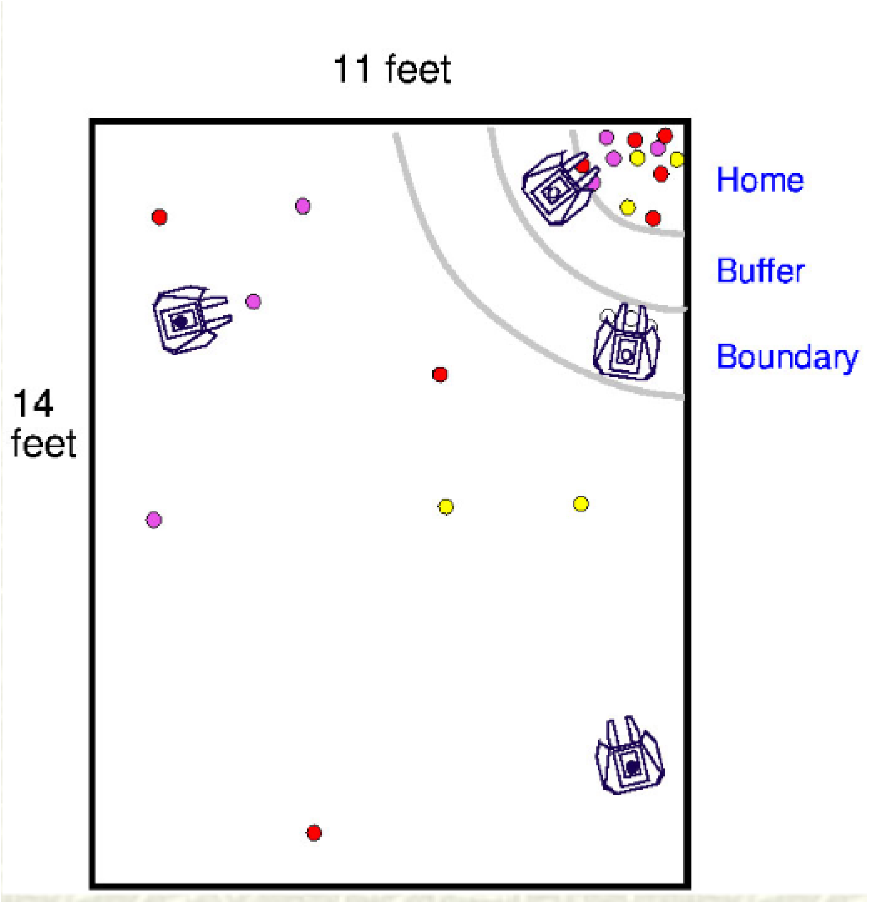
Solution: Reward shaping (intermediate rewards)
Add rewards/penalties for achieving subgoals/errors:
1. subgoal: grasped-puck
2. subgoal: dropped-puck-at-home
3. error: dropped-puck-away-from-home

$$\underline{r^*(s,a,s') = r(s,a,s') + F(s,s',\ldots)}$$

where F is called reward shaping

Reward Shaping will **speed up** the training process.(It is easy to understand.)

Example: Foraging Robots. Problems of Reward Shaping



Policy Invariant: Add Reward Shaping will not change the original policy

Reward shaping does not aim to **change** the ultimate policy but **help** the agent learn the optimal policy quickly. Policy invariant shaping **avoids** reward shaping that **misleads the agents**.

From formulation：
Q-Learning:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q_i(s',a') - Q(s,a)]$$

Q-Learning with reward shaping:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + F(s,s') + \gamma \max_{a'} Q_i(s',a') - Q(s,a)]$$

Potential-Based Reward Shaping

Ng, Russell and Harada. "Policy Invariance Under Reward Transformations: Theory And Application To Reward Shaping." ICML, 1999.

$$F(s, s') = \gamma\Phi(s') - \Phi(s)$$

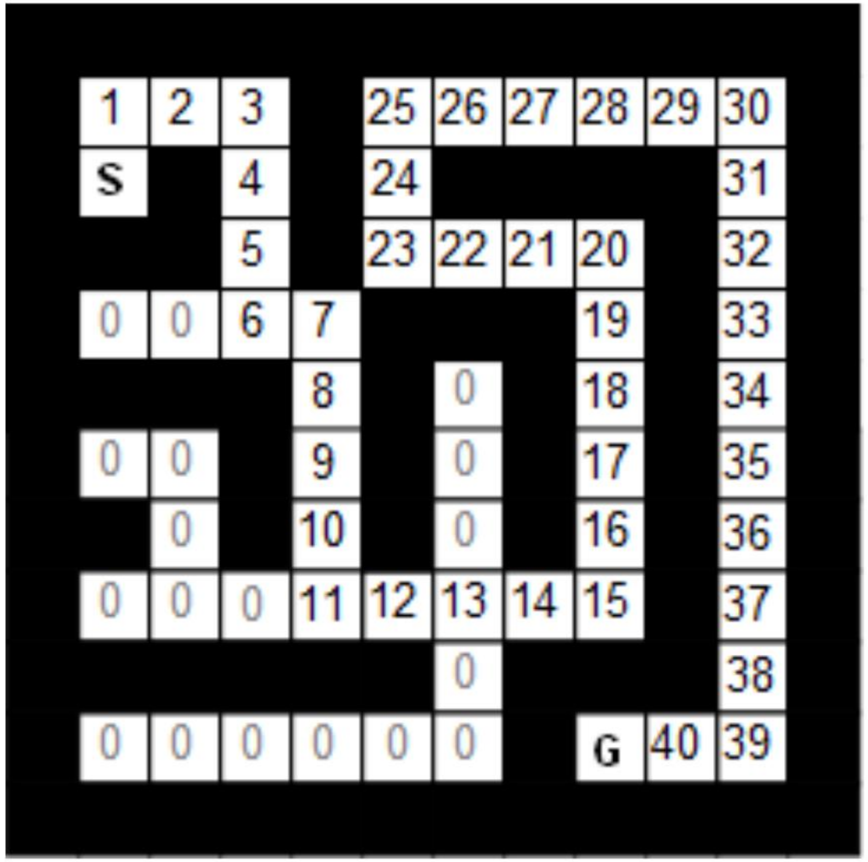
$F(s, s')$: Additional reward when moving from states to s'

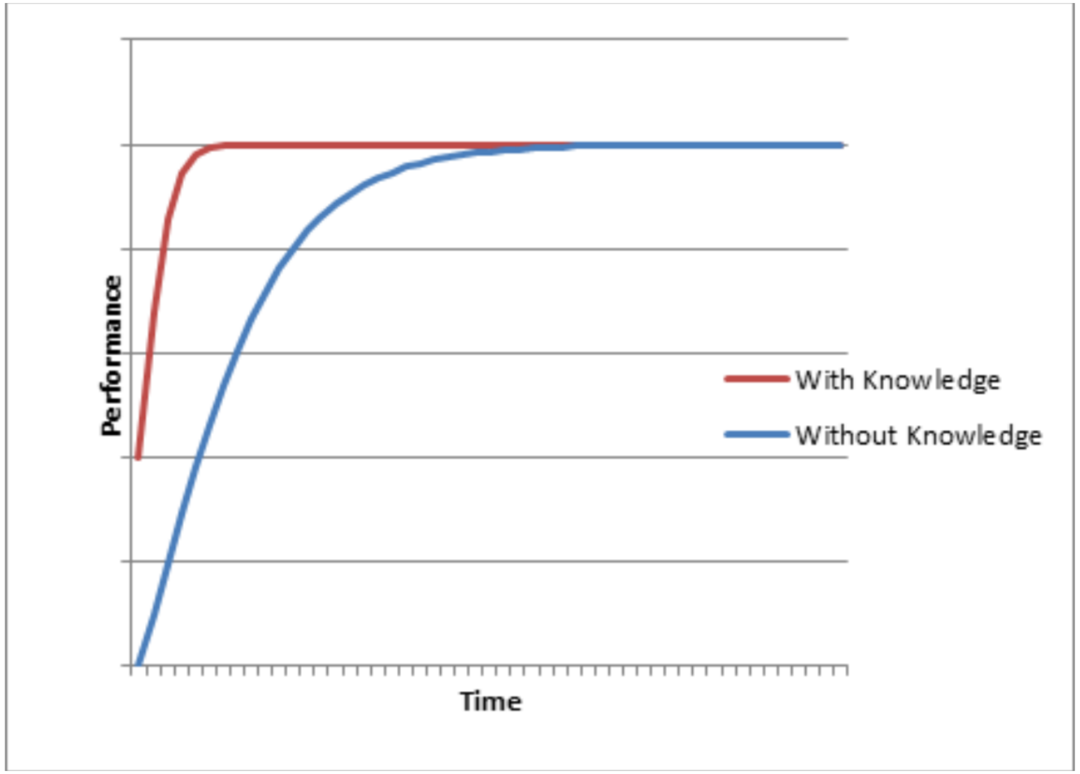
γ: Discount factor

$\Phi(s)$: Potential of states

Potential-Based Reward Shaping is Policy Invariant

Potential-Based Reward Shaping: Example





Potential-Based Reward Shaping

- Sufficiency Theorem: If F is a potential-based shaping function, then it is policy invariant.
  - Proof of Policy Invariant:
    - M is the original MDP; M' is the original MDP plus the shaping function $F(s, a, s') = \gamma\phi(s') - \phi(s)$
    - Claim 1: Any policy that optimizes $Q^*_{M'}(s, a)$ also optimizes $Q^*_M(s, a)$
    - Claim 2: Any policy that optimizes $Q^*_M(s, a)$ also optimizes $Q^*_{M'}(s, a)$
    - From claims 1 and 2, any optimal policy for M is also an optimal policy for M', and vice-versa. Therefore, F is policy invariant.

Potential-Based Reward Shaping

If F is a potential-based shaping function, then it is policy invariant.

Claim 1: Any policy that optimizes $Q^*_{M'}(s,a)$ also optimizes $Q^*_M(s,a)$.

- Begin with the Bellman equation for the optimal Q-function for M.

$$Q^*_M(s,a) = \sum_{s'} P_{sa}(s')\left[r(s,a,s') + \gamma \max_{a'\in A} Q^*_M(s',a')\right]$$

$$Q^*_M(s,a) - \Phi(s) = \sum_{s'} P_{sa}(s')\left[r(s,a,s') + \gamma \max_{a'\in A} Q^*_M(s',a')\right] - \Phi(s)$$

$$= \sum_{s'} P_{sa}(s')\left[r(s,a,s') - \Phi(s) + \gamma \max_{a'\in A} Q^*_M(s',a')\right]$$

$$= \sum_{s'} P_{sa}(s')\left[r(s,a,s') + \gamma\Phi(s') - \Phi(s) + \gamma \max_{a'\in A}\left[Q^*_M(s',a') - \Phi(s')\right]\right]$$

$$= \sum_{s'} P_{sa}(s')\left[r(s,a,s') + F(s,a,s') + \gamma \max_{a'\in A}\left[Q^*_M(s',a') - \Phi(s)\right]\right]$$

$$= \sum_{s'} P_{sa}(s')\left[r'(s,a,s') + \gamma \max_{a'\in A}\left[Q^*_M(s',a') - \Phi(s)\right]\right]$$

⌗ Above is the Bellman equation for $Q^*_{M'}(s,a)$, if we substitute $Q^*_{M'} = Q^*_M - \Phi(s)$:

$$Q^*_M(s,a) - \Phi(s) = \sum_{s'} P_{sa}(s')\left[r'(s,a,s') + \gamma \max_{a'\in A}\left[Q^*_M(s',a') - \Phi(s)\right]\right]$$

$$Q^*_{M'}(s,a) = \sum_{s'} P_{sa}(s')\left[r'(s,a,s') + \gamma \max_{a'\in A}\left[Q^*_{M'}(s,a)\right]\right]$$

⌗ Therefore, any policy that optimizes $Q^*_{M'}(s,a)$ also optimizes $Q^*_{M'} = Q^*_M - \Phi(s)$. Since Φ(s) does not depend on the action chosen in state s, this policy also maximizes $Q^*_M(s,a)$. That is, an optimal policy for M' is also an optimal policy for M.

Potential-Based Reward Shaping

Claim 2: Any policy that optimizes $Q^*_M(s,a)$ also optimizes $Q^*_{M'}(s,a)$

Begin with the Bellman equation for the optimal Q-function for *M'*.

$$Q^*_{M'}(s,a)=\sum_{s'}P_{sa}(s')\left[r'(s,a,s')+\gamma\max_{a'\in A}Q^*_{M'}(s',a')\right]$$

$$Q^*_{M'}(s,a)+\Phi(s)=\sum_{s'}P_{sa}(s')\left[r'(s,a,s')+\gamma\max_{a'\in A}Q^*_{M'}(s',a')\right]+\Phi(s)$$

$$=\sum_{s'}P_{sa}(s')\left[r'(s,a,s')+\Phi(s)+\gamma\max_{a'\in A}Q^*_{M'}(s',a')\right]$$

$$=\sum_{s'}P_{sa}(s')\left[r'(s,a,s')-\gamma\Phi(s')+\Phi(s)+\gamma\max_{a'\in A}(Q^*_{M'}(s',a')+\Phi(s'))\right]$$

$$=\sum_{s'}P_{sa}(s')\left[r'(s,a,s')-F(s,a,s')+\gamma\max_{a'\in A}(Q^*_{M'}(s',a')+\Phi(s))\right]$$

$$=\sum_{s'}P_{sa}(s')\left[r(s,a,s')+\gamma\max_{a'\in A}(Q^*_{M'}(s',a')+\Phi(s))\right]$$

Above is the Bellman equation for $Q^*_M(s,a)$, if we substitute $Q^*_M=Q^*_{M'}+\Phi(s)$:

$$Q^*_{M'}(s,a)+\Phi(s)=\sum_{s'}P_{sa}(s')\left[r(s,a,s')+\gamma\max_{a'\in A}(Q^*_{M'}(s',a')+\Phi(s))\right]$$

$$Q^*_M(s,a)=\sum_{s'}P_{sa}(s')\left[r(s,a,s')+\gamma\max_{a'\in A}(Q^*_M(s,a))\right]$$

Therefore, any policy that optimizes $Q^*_M(s,a)$ also optimizes $Q^*_M=Q^*_{M'}+\Phi(s)$. Since $\Phi(s)$ does not depend on the action chosen in state *s*, this policy also maximizes $Q^*_{M'}(s,a)$. That is, an optimal policy for *M* is also an optimal policy for *M'*.

Potential-Based Reward Shaping

- Choosing a potential function

If $\Phi(s) = V_M^*(s)$ then $V_{M'}^* = V_M^* - \Phi(s) = V_M^* - V_M^* = 0$

This form makes learning easy because the recursion in the Q-equations has been removed:

$$Q_M^*(s,a) = \sum_{s'} P_{sa}(s')\left[r(s,a,s') + \gamma V_M^*(s')\right]$$
$$= \sum_{s'} P_{sa}(s')\left[r(s,a,s') + 0\right]$$

Without knowing the actual value of $V_M^*(s,a)$ we can use our knowledge about the domain to estimate $V_M^*(s,a)$.

Further works

- Can be divided into these part:
  - 1. Representation of Shaping Function
  - 2. Learning the reward shaping(Choosing a potential function)
  - 3. Further Application(Policy Transfer)

Dynamic Potential-Based Reward Shaping.

Devlin and Kudenko. "Dynamic Potential-Based Reward Shaping." AAMAS, 2012.

- Expand the Potential-Based Reward Shaping
  - $F(s, s') = \gamma\phi(s') - \phi(s)$
  - $F(s, t, s', t') = \gamma\phi(s', t') - \phi(s, t)$

Principled methods for advising reinforcement learning agents

Wiewiora, Cottrell and Elkan. "Principled methods for advising reinforcement learning agents." ICML, 2003.

Look-Ahead Advice

- $F(s, a, s', a') = \gamma\Phi(s', a') - \Phi(s, a)$
- $\pi(s) = argmax_a\{Q(s, a) + \Phi(s, a)\}$
- Maintains all previous guarantees

Look-Back Advice

- $F(s, a, s', a') = \Phi(s', a') - \gamma^{-1}\Phi(s, a)$
- No guarantees proven

Potential-Based Reward Shaping forPartially Observable Markov Decision Processes.

Eck, Soh, Devlin and Kudenko. "Potential-Based Reward Shaping forPartially Observable Markov Decision Processes." AAMAS, 2013.

- Expand the Potential-Based Reward Shaping
  - $F(s, s') = \gamma\phi(s') - \phi(s)$
  - $F(o, o') = \gamma\phi(o') - \phi(o)$

Automatic shaping and decomposition of reward function

Marthi, Bhaskara. "Automatic shaping and decomposition of reward functions." international conference on machine learning (2007): 601-608.

Algorithm 1 Potential function learner. z is a state abstraction function, O is a set of options, and T is a nonnegative integer. The update procedure on line 9 maintains a simple running average, and assumes that unseen state–action pairs lead to a dummy terminal state with a very negative reward.

1: **function** LEARN-POTENTIAL-FUNCTION(z, O, T)
2:     Initialize transition, reward estimates $\hat{P}, \hat{R}$
3:     **repeat**
4:         $s \leftarrow$ current environment state
5:         Sample o randomly from O
6:         Follow option o until it terminates
7:         $s' \leftarrow$ current environment state
8:         $r \leftarrow$ be the total reward received while doing o
9:         Update $\hat{P}, \hat{R}$ using sample $(z(s), o, r, z(s'))$
10:     **until** T actions have been taken in the environment
11:     Solve MDP $\hat{\mathcal{M}} = (z(S), O, \hat{P}, \hat{R})$ exactly
12:     **return** value function of $\hat{\mathcal{M}}$
13: **end function**

Algorithm 2 Reward decomposition learner. z is a state abstraction function, O is a set of options, T, is a nonnegative integer, each f_e is a function from abstract states to a feature vector, and each g_e is a function from pairs (s, a) to a feature vector.

1: **function** LEARN-REWARD-DECOMPOSITION(z, O, T, $\{f_e\}$, $\{g_e\}$)
2:     Learn abstract MDP $\hat{\mathcal{M}}$ as in Algorithm 1 using z, O, T.
3:     Solve $\hat{\mathcal{M}}$, using linear approximation with the features in each f_e. Let α_e be the weight vector corresponding to f_e.
4:     Use the samples from step 2 to get a linear least squares estimate of the original MDP reward function in terms of the g_e. Let β_e be the weight vector corresponding to g_e.
5:     **Return** weights (α, β) corresponding to reward components $R_e(s, a, s') = \beta_e \cdot g_e(s, a) + \alpha_e \cdot (f_e(s') - f_e(s))$.
6: **end function**

Belief Reward Shaping in Reinforcement Learning

Marom O, Rosman B. Belief Reward Shaping in Reinforcement Learning[C]. national conference on artificial intelligence, 2018: 3762-3769.

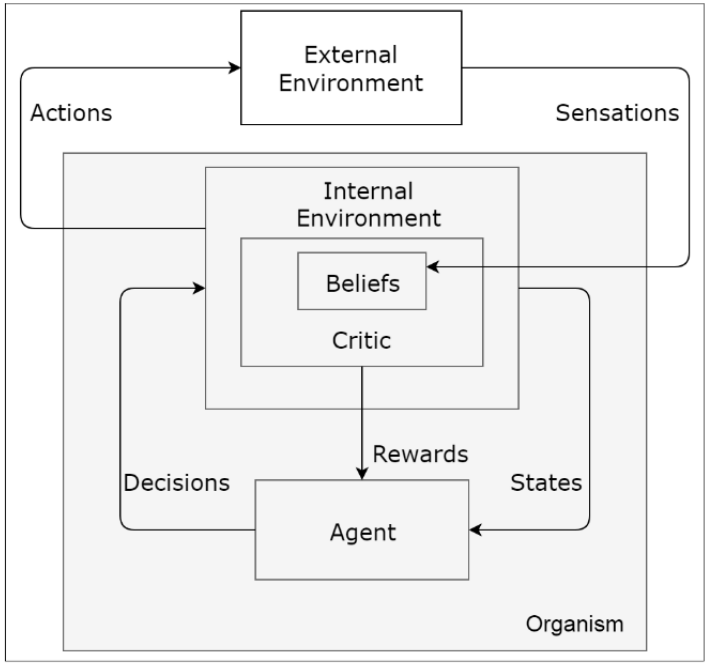


Figure 1: Agent / environment relationship with BRS.

Algorithm 1: Q-learning algorithm augmented with BRS for episodic tasks.

Initialise $Q(s,a)$ for all $s \in S$ and $a \in A_s$ and define $Q(s,a) = 0$ if s is terminal
For each $\tau \in \mathrm{T}$ select $F^\tau = \{\hat{p}^\tau(r|h) : h \in \mathcal{H}^\tau\}$ and $q^\tau(h)$
foreach *episode* **do**
  Initialise s
  while *s is not terminal* **do**
    Choose a from A_s using policy from Q
    Take action a. Observe next state s' and environment reward r
    Update $q^\tau(h|D^\tau)$ using equation 1
    Compute h^τ_{MAP} from $q^\tau(h|D^\tau)$
    Generate an unbiased estimate, $\hat{\mu}^B$, of $\mathbf{E}_{\hat{p}^\tau(r|h^\tau_{MAP})}[R]$
    $Q(s,a) \leftarrow Q(s,a) + \alpha(s,a)[\hat{\mu}^B + \gamma \sup_{b \in A_{s'}} Q(s',b) - Q(s,a)]$
    $s \leftarrow s'$
  end
end

Policy Transfer using Reward Shaping

Brys T, Harutyunyan A, Taylor M E, et al. Policy Transfer using Reward Shaping[J]. adaptive agents and multi-agents systems, 2015: 181-188.

- Transfer Learning:
    - $\chi S(s_{target}) = s_{source}$
    - $\chi A(a_{target}) = a_{source}$

In order to use the technique, we need to define a reward function R^π in the new task that captures the policy π transferred from the source task. The idea is to reward the learning agent for taking action a in state s, proportionally to the probability of the mapped state-action pair $(\chi_S(s), \chi_A(a))$ in the transferred policy:

$$R^\pi(s, a, s') = \pi(\chi_S(s), \chi_A(a)) \qquad (1)$$

Even though the formulation works for stochastic as well as deterministic policies, in this paper, we only focus on the latter. Therefore, R^π will always be either 0 or 1.

The negation of this reward function is then learned in a secondary value function Φ^π, whose values are used to shape the main reward R:

$$\begin{aligned} R_F(s, a, s', a') &= R(s, a, s') + F^\pi(s, a, t, s', a', t') \\ F^\pi(s, a, t, s', a', t') &= \gamma \Phi^\pi(s', a', t') - \Phi^\pi(s, a, t) \end{aligned}$$

Note that a simpler approach to policy transfer using shaping could be taken, using a static potential function:

$$\Phi(s, a) = \pi(\chi_S(s), \chi_A(, a)) \qquad (2)$$

Other works

- 1. Using Reward Shaping in Hierarchical Reinforcement Learning
  - Gao Y, Toni F. Potential based reward shaping for hierarchical reinforcement learning[C]. international conference on artificial intelligence, 2015: 3504-3510.

- 2. Learning Shaping through Inverse Reinforcement Learning
  - Suay H B, Brys T, Taylor M E, et al. Learning from Demonstration for Shaping through Inverse Reinforcement Learning[J]. adaptive agents and multi-agents systems, 2016: 429-437.

Other works

- Multi-Agent Potential-Based Reward Shaping
  - Devlin and Kudenko, "Theoretical Considerations Of Potential-BasedReward Shaping For Multi-Agent Systems", AAMAS, 2011.
- Main Difference:
  - Guarantees:
    - Nash Equilibria not altered
  - Can:
    - Increase/Decrease time taken to reach a stable joint policy
    - Change final joint policy