Reinforcement Learning

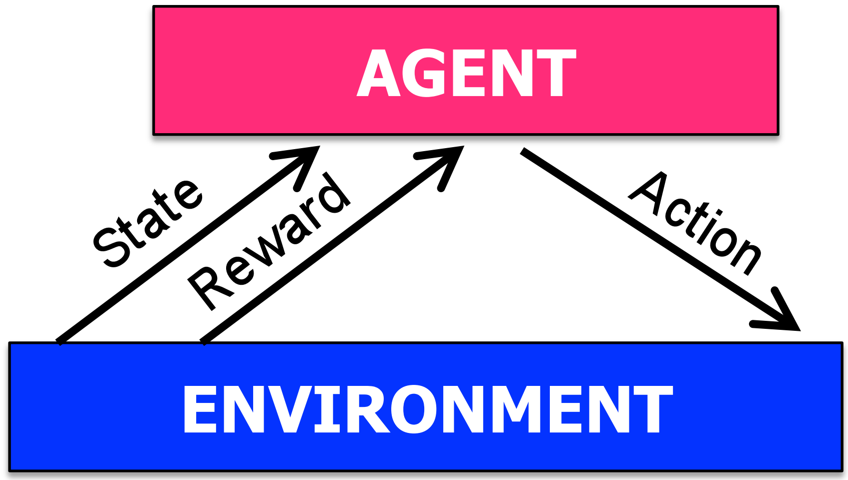
CS 550: Machine Learning

Reinforcement Learning

- Reinforcement learning addresses the question of how an autonomous agent that senses and acts in its environment can learn to choose optimal set of actions (learn a control policy) to achieve its goal

Reinforcement Learning

- Each time, after the agent takes an action, it MAY be provided with a reward (or a penalty), depending on the desirability of the next step that its action produces



- The task is to learn, from these indirect-delayed rewards, a policy (how to choose sequences of actions) that yields the greatest cumulative reward

$$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} \ldots$$

Goal is to learn a policy that maximizes the cumulative reward

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

γ determines the relative value of the delayed rewards to the immediate reward

Learning Task

- S is a set of states of the environment
- A is a set of actions that an agent can perform
- $r_t = r(s_t, a_t)$ is a reward/penalty that the environment gives at time t when the agent is at the state of s_t and it takes the action of a_t
- $s_{t+1} = \delta(s_t, a_t)$ is the succeeding state

The task is to learn a policy $\pi : S \rightarrow A$

that is to learn $\pi(s_t) = a_t$

– In a first order Markov decision process, the reward and succeeding state functions depend on only the current state and the current action

– These could also be nondeterministic functions

Learning Task

Select the policy that yields the greatest reward

Let $V^{\pi}(s_t)$ be the cumulative reward value achieved by an arbitrary policy π from an arbitrary initial state s_t

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$$

$0 \le \gamma < 1$ is a constant that determines the relative value of the delayed rewards to the immediate reward

$\gamma = 0 \Rightarrow$ only the immediate reward is considered

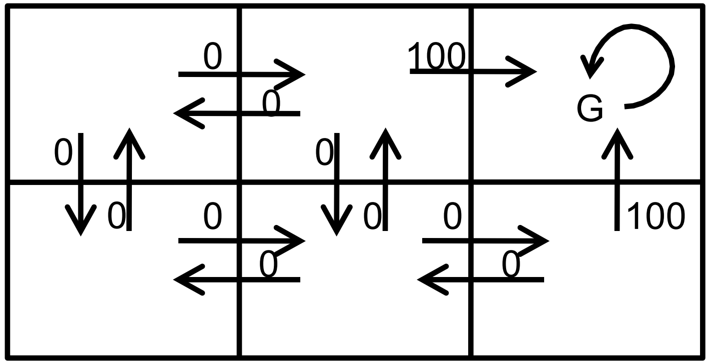
$\gamma \to 1 \Rightarrow$ future rewards are given greater emphasis

Learning Task

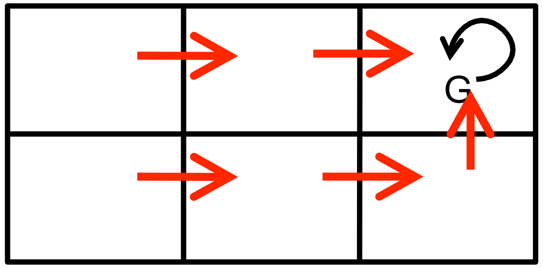
Optimal policy $\pi^* \equiv \arg\max_{\pi} V^{\pi}(s)$

$$\pi^*(s) = \arg\max_{a} \left[r(s,a) + \gamma V^*(\delta(s,a)) \right]$$

we use V^* instead of V^{π^*}



Consider the grid world above, where each entry corresponds to a state, arrows indicate the possible actions, numbers on the arrows are immediate rewards, and G is the goal (absorbing) state



One optimal policy

90	100	0
81	90	100

V*(s) values, Y=0.9

Learning Task

Optimal policy $\pi^* \equiv \arg\max_{\pi} V^{\pi}(s)$

$$\pi^*(s) = \arg\max_{a} \left[r(s,a) + \gamma V^*(\delta(s,a)) \right]$$

we use V^*

instead of V^{π^*}

However, V is defined on the states not on the actions. Thus, we will introduce the Q-function which is defined on the actions*

$$Q(s,a) \equiv r(s,a) + \gamma V^*(\delta(s,a))$$

$$\pi^*(s) = \arg\max_{a} Q(s,a)$$

$$V^*(s) = \max_{a'} Q(s,a')$$

$$Q(s,a) = r(s,a) + \gamma \max_{a'} Q(\delta(s,a),a')$$

Q-Learning Algorithm

For each s and a, initialize $\hat{Q}(s,a) = 0$

Observe the current state s

Do forever

Select an action a and execute it

Receive the immediate reward r

Observe the new state s'

Update the table entry for $\hat{Q}(s,a) = r + \gamma \max_{a'} \hat{Q}(s',a')$

$s = s'$

Reinforcement Learning

- Delayed rewards
- Exploration versus exploitation
  - To select the action sequences in training
- Partially observed states
- Life-long learning