Active Reinforcement Learning

Difference with Passive RL

- A passive learning agent has a fixed policy that determines its behaviour. An active agent must decide what action to take
- An active agent requires outcome probabilities of ALL ACTIONS rather than for a fixed policy as used in passive RL
- An active agent EXPLORES the world
- Trade off between Exploration and Exploitation
  - Sticking to only known world ensures stability but may lead to sub-optimal solution
  - Exploring new opportunities lead to improve the present situation

Q-Learning

- Define a new function Q (a, s)
- Relationship with Utility: $U(s)=\max_a Q(a, s)$
- Constraint Equation for Equilibrium: $Q(a, S) = R(S) + \gamma \sum_{s'} T(s, a, s') \max_{a'} Q(a', s')$
- TD Update for Utility: $U^{\pi}(s) \leftarrow U^{\pi}(s) + \alpha(R(s) + \gamma U^{\pi}(s') - U^{\pi}(s))$
- Q learning with TD update: $Q(a, S) \leftarrow Q(a, S) + \alpha(R(S) + \gamma \max_{a'} Q(a', s') - Q(a, s)$

Q - Learning

Update after each state transition

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{current value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{current value}} \Big)}^{\text{temporal difference}}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

where r_t is the reward received when moving from the state s_t to the state s_{t+1}, and α is the learning rate $(0 < \alpha \leq 1)$.

Note that $Q^{new}(s_t, a_t)$ is the sum of three factors:

- $(1 - \alpha)Q(s_t, a_t)$: the current value (weighted by one minus the learning rate)
- $\alpha\, r_t$: the reward $r_t = r(s_t, a_t)$ to obtain if action a_t is taken when in state s_t (weighted by learning rate)
- $\alpha\gamma \max_a Q(s_{t+1}, a)$: the maximum reward that can be obtained from state s_{t+1} (weighted by learning rate and discount factor)

An episode of the algorithm ends when state s_{t+1} is a final or *terminal state*. However, Q-learning can also learn in non-episodic tasks (as a result of the property of convergent infinite series). If the discount factor is lower than 1, the action values are finite even if the problem can contain infinite loops.

For all final states s_f, $Q(s_f, a)$ is never updated, but is set to the reward value r observed for state s_f. In most cases, $Q(s_f, a)$ can be taken to equal zero.

Problem with Optimal Policy

Agent learns a model not the true environment !!

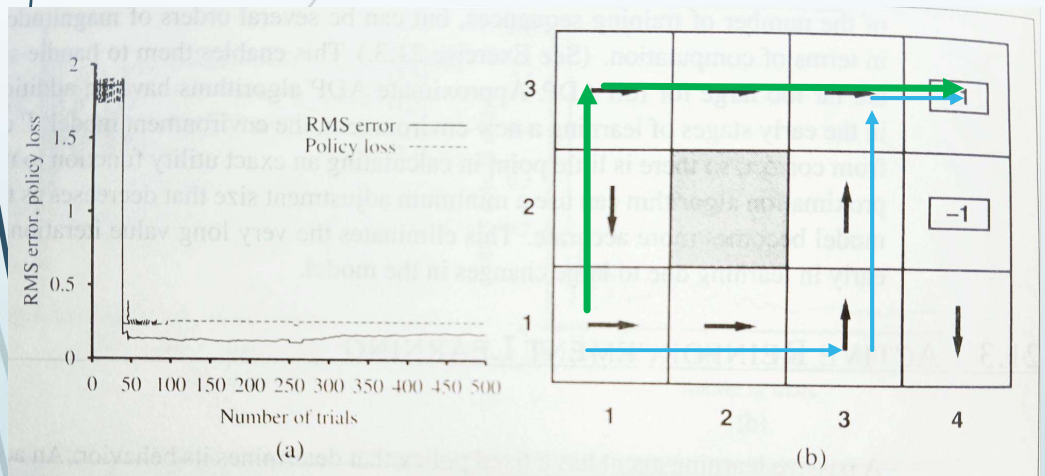

Figure 21.6 Performance of a greedy ADP agent that executes the action recommended by the optimal policy for the learned model. (a) RMS error in the utility estimates averaged over the nine nonterminal squares. (b) The suboptimal policy to which the greedy agent converges in this particular sequence of trials.

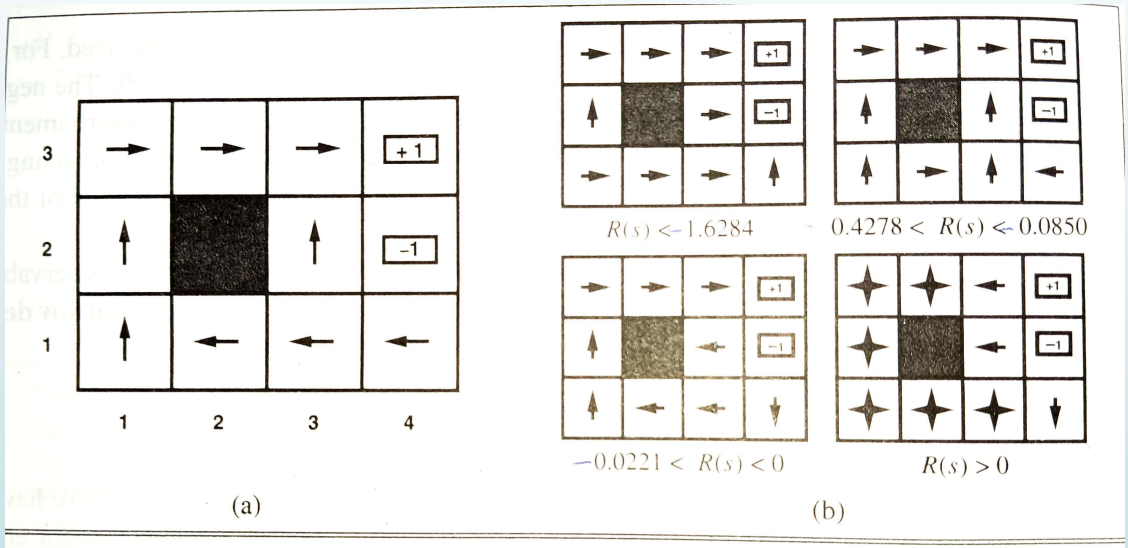

Figure 17.2 (a) An optimal policy for the stochastic environment with $R(s) = -0.04$ in the nonterminal states. (b) Optimal policies for four different ranges of $R(s)$.

Exploration Function

- Two new functions
- N(a, s) = How many times action a is executed at state s
- **Exploration function f (u, n)**
  - Increasing in u and decreasing in n
  - Greed is traded off with curiosity
  - $f(u, n) = \begin{cases} R^+ & \text{if } n < N_e \\ u & \text{otherwise} \end{cases}$
    - where R+ is an optimistic estimate of the best possible reward obtainable in any state and Ne is a fixed parameter
    - Ensures each state-action pair will be tried at least Ne times

Q – Learning implementation

function Q-LEARNING-AGENT(*percept*) **returns** an action
  inputs: *percept*, a percept indicating the current state s' and reward signal r'
  static: Q, a table of action values index by state and action
    N_{sa}, a table of frequencies for state-action pairs
    s, a, r, the previous state, action, and reward, initially null

  if s is not null **then do**
    increment $N_{sa}[s, a]$
    $Q[a, s] \leftarrow Q[a, s] + \alpha(N_{sa}[s, a])(r + \gamma \max_{a'} Q[a', s'] - Q[a, s])$
  if TERMINAL?$[s']$ **then** $s, a, r \leftarrow$ null
  else $s, a, r \leftarrow s', \text{argmax}_{a'} f(Q[a', s'], N_{sa}[a', s']), r'$
  return a

Figure 21.8 An exploratory Q-learning agent. It is an active learner that learns the value $Q(a, s)$ of each action in each situation. It uses the same exploration function f as the exploratory ADP agent, but avoids having to learn the transition model because the Q-value of a state can be related directly to those of its neighbors.

30 Bandit Problem

Can we have an optimal exploration function