ADVANCED MACHINE LEARNING

Discrete Reinforcement Learning
Summary
RL: Markov Decision Process (MDP)

MDP

\[ S = \{s_0, s_1, \ldots, s_{225}\} \]
\[ A = \{\text{up, down, left, right}\} \]
\[ P_{ss'}^a = p(s_{t+1}|a_t, s_t) \]
\[ R_{ss'}^a = p(r_{t+1}|a_t, s_t, s_{t+1}) \]
\[ \gamma \in \mathbb{R} \]
Model-based reinforcement learning

To estimate $V$, one needs a model of the world to estimate the state transitions and reward distribution if stochastic.

$$V^\pi(s) = \sum_a \pi(s,a) \sum_{s'} P_{ss'}^a \left[ R_{ss'}^a + \gamma V^\pi(s') \right]$$

If this is not known, one resort to sample based techniques.
The trio V-Q-P

Bellman recursion

\[ V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^\pi(s')] \]

\[ Q^\pi(s, a) = \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^\pi(s')] \]

\[ \pi(s, a) = \frac{\exp(Q^\pi(s, a)/t)}{\sum_i \exp(Q^\pi(s, a_i)/t)} \]
The trio V-Q-P

Bellman recursion

\[ V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^\pi(s')] \]

\[ Q^\pi(s, a) = \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^\pi(s')] \]

\[ \pi(s, a) = \frac{\exp(Q^\pi(s, a)/t)}{\sum_i \exp(Q^\pi(s, a_i)/t)} \]

\[ V^\pi(s) = \sum s_{16} P_{ss'}^a [R_{ss'}^a + \gamma V^\pi(s')] \times s_{16} \]

Repeat until convergence!
Bellman residual
RL: DP Summary

Policy evaluation and improvement

1. Evaluate policy: \( \pi(s, a) \) Policy evaluation is Linear

\[
V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^\pi(s')] \\
Q^\pi(s, a)
\]

2. Improve policy:

\[
\pi(s, a) = \frac{\exp(Q(s,a)/t)}{\sum_i \exp(Q(s,a_i)/t)} \quad t \to 0
\]

becomes greedy

Value Iteration

1. Evaluate policy & improve it: Policy evaluation is Non-linear

\[
V^*(s) = \max_a \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^\pi(s')] \\
R^a_{ss'}, \quad P^a_{ss'}
\]

We need to know the models!
Advanced Machine Learning

Value Iteration & Policy Evaluation

**Value Iteration**

**Policy Evaluation**

**Optimal value function:**

\[ V^*(s) = \max_a \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^*(s')] \]

**Evaluate random policy:**

\[ V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^\pi(s')] \]
Bellman residual

Stopping thresholds

\[ \max_s |V_{k+1}(s) - V_k(s)| \]
Policy Evaluation & Improvement

Generalized Policy Iteration

\[ V = V_{\pi} \]

\[ \pi = \text{greedy}(V) \]

Sutton & Barto Chapter 4
Value Iteration & Policy Evaluation (FINAL)

**Value Iteration**

*Optimal value function:*

\[
V^*(s) = \max_a \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^*(s')] 
\]

**Policy Evaluation**

*Evaluate random policy:*

\[
V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a [R_{ss'}^a + \gamma V^\pi(s')] 
\]
Dynamic Programming

Dynamic Programming: $V(s_t) \leftarrow E_{\pi} \{ r_{t+1} + \gamma V(s_t) \}$
TD-Learning

Monte-Carlo: \( V(s_t) \leftarrow V(s_t) + \eta [R_t - V(s_t)] \)

where \( R_t \) is the actual return following state \( s_t \).

Adapted from R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction
TD-Learning

\[ V(s_t) \leftarrow V(s_t) + \eta \left[ r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \right] \]

Adapted from R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction
### Dynamic Programming: DP

- **No interaction with the environment**
- Offline
  - Bellman Equation
    - Policy Evaluation: $V^\pi(s)$
    - Value Iteration: $V^*(s)$
- We have the models!!!

### Monte Carlo: MC

- **Physical interaction with the environment**
- Online
- Physical interaction with the environment
  - Temporal difference (TD)
    - TD: $V^\pi(s)$
    - SARSA: $Q^\pi(s, a)$
    - Q-Learning: $Q^*(s, a)$
- To hard to model !!!

---

**Block World**

- State $s_t$
- Action $a_t$
- Reward $r_t$
- Next state $s_{t+1}$

---

**DP – MC - TD**
State Action Reward State Action

SARSA

\[ V^\pi(s) \leftarrow \alpha [r + \gamma V^\pi(s') - V^\pi(s)] \]

\[ Q^\pi(s, a) \leftarrow \alpha [r + \gamma Q^\pi(s', a') - Q^\pi(s, a)] \]