Reinforcement learning:
Examples and proofs

Francisco S. Melo
fmelo@isr.ist.utl.pt

Reading group on Sequential Decision Making
Outline of the presentation

- A simple problem
- Dynamic programming (DP)
- $Q$-learning
- Convergence of DP
- Convergence of $Q$-learning
- Further examples
A simple problem

Problem:
An autonomous robot must learn how to transport material from a deposit to a building facility.
The Markov decision process model

**Markov Decision Process:** \((S, A, P, r)\)

- **States:** \(S = \{1_U, 2_U, 3_U, 1_L, 2_L, 3_L\}\);
  - \(1_U\) Robot in position 1 (unloaded);
  - \(2_U\) Robot in position 2 (unloaded);
  - \(3_U\) Robot in position 3 (unloaded);
  - \(1_L\) Robot in position 1 (loaded);
  - \(2_L\) Robot in position 2 (loaded);
  - \(3_L\) Robot in position 3 (loaded)

- **Actions:** \(A = \{\text{Left, Right, Load, Unload}\}\);
The Markov decision process model (2)

- Transition probabilities: “Left”/“Right” move the robot in the corresponding direction; “Load” loads material (only in position 1); “Unload” unloads material (only in position 3).

Ex:

\[(2_L, \text{Right}) \rightarrow 3_L;\]
\[(3_L, \text{Unload}) \rightarrow 3_U;\]
\[(1_L, \text{Unload}) \rightarrow 1_L.\]

- Reward: We assign a reward of $+10$ for every unloaded package (payment for the service).
Outline of the presentation

• A simple problem
• Dynamic programming (DP)
• $Q$-learning
• Convergence of DP
• Convergence of $Q$-learning
• Some more examples
Dynamic programming

- For each action $a \in A$, $P_a$ is a matrix.

Ex:

$$P_{\text{Right}} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}$$
Dynamic programming (2)

- The reward $r(s, a, s')$ can also be represented as a matrix

Ex:

$$r(\cdot, a, \cdot) = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & +10 & 0 & 0 & 0 \\
\end{bmatrix}$$
Dynamic programming (3)

Recall that

$$Q^*(s, a) = \sum_{s' \in S} P_a(s, s') [r(s, a, s') + \gamma \max_{b \in A} Q^*(s', b)].$$

From $Q^*$ we can compute the optimal policy $\pi^*$:

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

and the optimal value function

$$V^*(s) = \max_{a \in A} Q^*(s, a).$$
Dynamic programming (4)

Since $S$ and $A$ are finite, $Q^*(s, a)$ is a matrix.

Iterations of DP:

$$Q_0 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}$$

$$Q_1 = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 10 \\
\end{bmatrix}$$
Dynamic programming (5)

Iterations of DP:

\[ Q_5 = \begin{bmatrix} 0 & 0 & 8.57 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 8.57 & 9.03 & 8.57 & 8.57 \\ 8.57 & 9.5 & 9.03 & 9.03 \\ 9.03 & 9.5 & 9.5 & 10 \end{bmatrix} \quad Q_{20} = \begin{bmatrix} 18.53 & 17.61 & 19.51 & 18.54 \\ 18.53 & 16.73 & 17.61 & 17.61 \\ 17.61 & 16.73 & 16.73 & 16.73 \\ 19.51 & 20.54 & 19.51 & 19.51 \\ 19.51 & 21.62 & 20.54 & 20.54 \\ 20.54 & 21.62 & 21.62 & 26.73 \end{bmatrix} \]
### Dynamic programming (6)

Final $Q^*$ and policy:

$$Q^* = \begin{bmatrix}
30.75 & 29.21 & 32.37 & 30.75 \\
30.75 & 27.75 & 29.21 & 29.21 \\
29.21 & 27.75 & 27.75 & 27.75 \\
32.37 & 34.07 & 32.37 & 32.37 \\
32.37 & 35.86 & 34.07 & 34.07 \\
34.07 & 35.86 & 35.86 & 37.75 \\
\end{bmatrix}$$

$$\pi^* = \begin{bmatrix}
\text{Load} \\
\text{Left} \\
\text{Left} \\
\text{Right} \\
\text{Right} \\
\text{Unload} \\
\end{bmatrix}$$
Outline of the presentation

- A simple problem
- Dynamic programming (DP)
- $Q$-learning
- Convergence of DP
- Convergence of $Q$-learning
- Some more examples
Once again,

\[
Q^*(s, a) = \sum_{s' \in S} P_a(s, s') [r(s, a, s') + \gamma \max_{b \in \mathcal{A}} Q^*(s', b)] = \\
= \mathbb{E} \left[ r(s, a, s') + \gamma \max_{b \in \mathcal{A}} Q^*(s', b) \right].
\]

\(Q\)-learning approximates the expectation above by point-samples: given transition triplets \((s, a, s', r)\) sampled from the MDP, \(Q\)-learning follows the update rule

\[
Q_{t+1}(s, a) = Q_t(s, a) + \alpha_t(s, a) \left( r + \gamma \max_{b \in \mathcal{A}} Q_t(s', b) - Q_t(s, a) \right).
\]
Outline of the presentation

• A simple problem
• Dynamic programming (DP)
• $Q$-learning
• Convergence of DP
• Convergence of $Q$-learning
• Some more examples
Convergence of DP

Given a general function \( q : S \times A \rightarrow \mathbb{R} \), define the operator \( H \) as

\[
(Hq)(s, a) = \sum_{s' \in S} P_a(s, s') \left[ r(s, a, s') + \gamma \max_{b \in A} q(s', b) \right].
\]

This operator is a contraction in the norm \( \| \cdot \|_\infty \):

\[
\|Hq_1 - Hq_2\|_\infty \leq \gamma \|q_1 - q_2\|_\infty.
\]
Convergence of DP (2)

$Q^*$ is a vector in $\mathbb{R}^{|S| \times |A|}$, a complete metric space endowed with the metric $d(q_1, q_2) = \|q_1 - q_2\|_\infty$. Then, convergence of DP is an immediate consequence of

**Theorem 1** (Banach fixed point theorem). Let $(X, d)$ be a non-empty complete metric space. Let $H : X \rightarrow X$ be a contraction mapping on $X$. Then the map $H$ admits one and only one fixed point $x^*$ in $X$ (this means $H(x^*) = x^*$). Furthermore, this fixed point can be found as follows: start with an arbitrary element $x_0 \in X$ and define an iterative sequence by $x_n = H(x_{n-1})$ for $n = 1, 2, 3, \ldots$. This sequence converges, and its limit is $x^*$. 
Outline of the presentation

- A simple problem
- Dynamic programming (DP)
- $Q$-learning
- Convergence of DP
- Convergence of $Q$-learning
- Some more examples
Convergence of Q-learning

Convergence of Q-learning uses the following simple convergence theorem:

**Theorem 2.** The random process \( \{\Delta_t\} \) in \( \mathbb{R}^n \) defined as

\[
\Delta_{t+1}(x) = (1 - \alpha_t(x))\Delta_t(x) + \alpha_t(x)F_t(x)
\]

converges to zero w.p.1 under the following assumptions:

- \( 0 \leq \alpha_t \leq 1 \), \( \sum_t \alpha_t(x) = \infty \) and \( \sum_t \alpha_t^2(x) < \infty \);
- \( \|\mathbb{E}[F_t(x) | \mathcal{F}_t]\|_W \leq \gamma \|\Delta_t\|_W \), with \( \gamma < 1 \);
- \( \text{var}[F_t(x) | \mathcal{F}_t] \leq C(1 + \|\Delta_t\|_W^2) \), for \( C > 0 \).