
Spring 2019

Reinforcement learning – part 1
Overview

• Probability basics
• Reinforcement learning
  – Markov Decision Process
  – Solving MDPs
• References
  – Reinforcement learning – Sutton & Barto, Second edition, 2018
How can we make [robots] to make [optimal] decisions?
Planning recap

• A*  
  – Admissible (underestimate) heuristics  
  – Consistent (monotonic) heuristics  

• Dynamic programming – recursion  
  \( V(x_t) \) as a function of \( V(x_{t+1}) \) or \( V(x_{t-1}) \)  

• D* dynamic repair
Exit from deterministic world
Enter stochastic world
Probability preliminaries

• Probability represents uncertainty
• $0 \leq P(A) \leq 1$
• $P(\text{True}) = 1$
• $P(\text{False}) = 0$
Probability preliminaries

- Probabilities over all events sum to 1
  \[ P(A) + P(\neg A) = 1 \]
- \( P(\neg A) = 1 - P(A) \)

\( p(A) \): Probability of \( A \) being true
\( p(\neg A) \): probability of \( A \) not being true
Probability preliminaries

- Probabilities over all events sum to 1

$$\sum_{v \in D(x)} P(x = v) = 1$$

Where $D(x)$ denotes the domain of valid values for variable $x$
\[ P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B) \]
Conditional probability

$p(A)$: Probability of A (given the world)

A: You will be in Greece next month.  \[ P(A) = 0.1 \]
Conditional probability

$p(A)$: Probability of A (given the world)

A: You will be in Greece next month.
B: Your paper got accepted at a conference to be held in Greece next month.

$P(A) = 0.1$
$P(B) = 0.05$
$P(A \cap B) = 0.04$
Conditional probability

$p(A)$: Probability of A (given the world)

A: You will be in Greece next month.
B: Your paper got accepted at a conference to be held in in Greece next month.

$P(A) = .1$
$P(B) = .05$
$P(A^B) = .04$

$P(A) + P(\sim A) = 1$
Conditional probability

\[ p(A|B): \text{Probability of A "given" B} \]

A: You will be in Greece next month.
B: Your paper got accepted at a conference to be held in Greece next month.

\[
p(A|B) = \frac{p(A \cap B)}{p(B)}
\]

\[
P(A) = 0.1 \\
P(B) = 0.05 \\
P(A \cap B) = 0.04 \\
P(A|B) = \frac{0.04}{0.05} = 0.8
\]

\[ P(A|B) + P(\sim A|B) = 1 \]
Conditional probability

\[ p(A|B): \text{Probability of } A \text{ "given" } B \]

\[ p(A \mid B) = \frac{p(A \wedge B)}{p(B)} \]

\[ p(A \wedge B) = p(A \mid B)p(B) \]

\[ p(A \wedge B) = p(B \mid A)p(A) \]

\[ p(A \mid B)p(B) = p(B \mid A)p(A) \]
Conditional probability

\( p(A|B): \text{Probability of } A \text{ "given" } B \)

\[
p(A | B) = \frac{p(A \cap B)}{p(B)}
\]

\[
p(A \cap B) = p(A | B) p(B)
\]

Bayes’ rule:

\[
p(A | B) = \frac{p(B | A) p(A)}{p(B)}
\]

\[
p(A \cap B) = p(B | A) p(A) = p(A | B) p(B)
\]

Conditional probability

\( P(B|A): \text{Probability of } B \text{ "given" } A \)

\[
p(B | A) = \frac{p(A \cap B)}{p(A)}
\]

\[
p(A \cap B) = p(B | A) p(A)
\]
If you mail me your stub I will pay you $100.

Expected value of mailing your stub = $100

$100 - $60 = $40
If you mail me your stub I will pay you $100.
If I don’t receive it I won’t pay anything.
Probability of mail getting lost is 0.5.

Expected value of mailing your stub = $100
$100 - $60 = $40
If you mail me your stub I will pay you $100.
If I don’t receive it I won’t pay anything.
Probability of mail getting lost is 0.5.
Expected value

- $X = x_1, \ldots, x_n$: set of $n$ possible outcomes
- $v(x_i)$: value associated with outcome $x_i$
- $p(x_i)$: $p(X=x_i)$ probability of outcome $x_i$; $\sum_{i=1}^{n} p(x_i) = 1$
- Expected value for a single value $X=x_i$: $p(x_i)v(x_i)$
Expected value

- $X = x_1, \ldots, x_n$: set of $n$ possible outcomes
- $v(x_i)$: value associated with outcome $x_i$
- $p(x_i)$: $p(X=x_i)$ probability of outcome $x_i$; $\sum_{i=1}^{n} p(x_i) = 1$
- Expected value for a single value $X=x_i$: $p(x_i)v(x_i)$
- Expected value of $X$:

$$E(X) = \sum_{i=1}^{n} p(x_i)v(x_i)$$
Questions on preliminaries?
Learning what to do to solve a problem

• e.g. how to open a door
Supervised learning

• Told what to do from someone who knows how to solve the problem
• Learning from examples
Reinforcement learning

• Interact with environment to learn what works and what doesn’t
What is the goal of reinforcement learning?
Policy

Prepare what to do to maximize long-term, possibly discounted, expected reward wherever you are.

Mapping from state to action(s)
• Deterministic policy: $\pi(s) \rightarrow a$
• Stochastic policy: $\pi(s,a) = \pi(als) \rightarrow p(als)$ for each available action $a$
Reinforcement learning

• Reward function: immediate reward of state
• Value function: long-term reward of state
• Policy: a mapping from state to action(s)
• Find a policy that maximizes long-term reward
Markov property

It doesn’t matter how you got here, but whatever happens from now on depends only on your current state.

\[ p(s_{t+1} = s', r_{t+1} = r \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_0, a_0) = p(s_{t+1} = s', r_{t+1} = r \mid s_t, a_t) \]
Reward vs. utility

- Decision theory
- Reward: face value
- Utility: relative value based on which people make decisions
- e.g., the second piece of cheesecake or $100K today vs. $100K in 10 years
- Discounted rewards
Short-term vs. long-term benefits

- Decision making based on:
  - Immediate rewards
  - Long-term rewards
  - Long-term expected rewards
  - Discounted long-term expected rewards
Stochastic state transition

• Deterministic

\[
p(s_{i+1} | s, a) = 1 \quad \text{for } i = 1, \ldots, n
\]

\[
\sum_{i=1}^{n} p(s_i | s, a) = 1
\]

• Nondeterministic

\[
p(s_1 | s, a) \quad s_1 \quad r(s_1)
\]

\[
p(s_2 | s, a) \quad s_i \quad r(s_i)
\]

\[
p(s_n | s, a) \quad s_n \quad r(s_n)
\]
Markov Decision Process

- MDP = \{ S, A, T, r, \gamma \}
- S: set of states
- A: set of actions
- T: state transition matrix \( S \times A \times S \rightarrow \mathbb{R} \)
- r: reward function \( S \times A \rightarrow \mathbb{R} \)
- \( \gamma \): discount factor \((0,1]\)
Value function

\[ V^\pi(s) = E_\pi \{ R_t \mid s_t = s \} \]
\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

State-value function for policy \( \pi \): expected return of starting at state \( s \) and follow policy \( \pi \) thereafter.

\[ Q^\pi(s, a) = E_\pi \{ R_t \mid s_t = s, a_t = a \} \]
\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right\} \]

Action-value function for policy \( \pi \)
State-value function

\[ V^\pi(s) = E_\pi \{ R_t \mid s_t = s \} \]
\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

State-value function for policy \( \pi \): expected return of starting at state \( s \) and follow policy \( \pi \) thereafter.

Backup diagram
Action-value function

\[ Q^\pi(s, a) = E_\pi \{ R_t | s_t = s, a_t = a \} \]

\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a \right\} \]

Action-value function for policy \( \pi \): expected return of starting at state \( s \), taking action \( a \), and follow policy \( \pi \) thereafter.

Backup diagram

CMU 16-785: Integrated Intelligence in Robotics
Jean Oh 2019
Which action should we take?

\[ V(s) \quad Q(s, a) \]
Bellman equation of value function

\[ V^\pi(s) = E_\pi \{ R_t \mid s_t = s \} \]

\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

\[ = E_\pi \left\{ r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \right\} \]

\[ = r_{t+1} + \gamma (r_{t+2} + \gamma r_{t+3} + \ldots) \]
Bellman equation of value function

\[ V^\pi(s) = E_\pi \{ R_t \mid s_t = s \} \]

\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

\[ = E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\} \]

\[ = r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{(t+1)k+1} \]
Bellman equation of value function

\[ V^\pi(s) = E_\pi \{ R_t \mid s_t = s \} \]

\[ = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

\[ = E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\} \]
Bellman equation of value function

\[ V^\pi(s) = E_{\pi} \{ R_t \mid s_t = s \} \]

\[ = E_{\pi} \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

\[ = E_{\pi} \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\} \]

\[ = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma E_{\pi} \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_{t+1} = s' \right\} \right] \]
Bellman equation of value function

\[
V^{\pi}(s) = E_{\pi}\{ R_t \mid s_t = s \}
= E_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\}
= E_{\pi}\left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\}
= \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma E_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+1+k+1} \mid s_{t+1} = s' \right\} \right]
= \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^{\pi}(s') \right]
\]
Optimal policy

- $\pi \succeq \pi'$ iff. $V^\pi(s) \geq V^{\pi'}(s)$ for every state $s$
- Optimal policies $\pi^*$ share optimal state-value function $V^*$ (or action-value function $Q^*$)

\[
V^*(s) = \max_{\pi} V^\pi(s), \forall s \in S \\
Q^*(s,a) = \max_{\pi} Q^\pi(s,a), \forall s \in S, a \in A \\
Q^*(s,a) = E\left\{r_{t+1} + \gamma V^*(s_{t+1}) \bigg| s_t = s, a_t = a\right\}
\]
Bellman optimality equation for $V^*$

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

$$= \max_a E_{\pi^*} \left\{ R_t \mid s_t = s, a_t = a \right\}$$

$$= \max_a E_{\pi^*} \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right\}$$

$$= \max_a E_{\pi^*} \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s, a_t = a \right\}$$

$$= \max_a E_{\pi^*} \left\{ r_{t+1} + \gamma V^* (s_{t+1}) \mid s_t = s, a_t = a \right\}$$

$$= \max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^*(s') \right]$$
Bellman optimality equation for $Q^*$

$$Q^*(s,a) = E\left\{ r_{t+1} + \gamma \max_a Q^*(s_{t+1},a') \mid s_t = s, a_t = a \right\}$$

$$= \sum_{s'} p(s' \mid s,a) \left[ E(r_{t+1} \mid s,a,s') + \gamma \max_a Q^*(s',a') \right]$$
Bellman optimality equations

\[ V^* (s) = \max_a \mathbb{E}^\pi \left\{ r_{t+1} + \gamma V^* (s_{t+1}) \mid s_t = s, a_t = a \right\} \]
\[ = \max_a \sum_{s'} p(s' \mid s, a) \left[ \mathbb{E}(r_{t+1} \mid s, a, s') + \gamma V^*(s') \right] \]

\[ Q^*(s, a) = \mathbb{E} \left\{ r_{t+1} + \gamma \max_a Q^*(s_{t+1}, a') \mid s_t = s, a_t = a \right\} \]
\[ = \sum_{s'} p(s' \mid s, a) \left[ \mathbb{E}(r_{t+1} \mid s, a, s') + \gamma \max_a Q^*(s', a') \right] \]
Solving MDP

• Have a probability model
• Dynamic programming
  – Policy iteration
  – Value iteration
Backward value iteration

\[
G^*_k(x_k) = \min_{u_k, \ldots, u_K} \left\{ \sum_{i=k}^{K} l(x_i, u_i) + l_F(x_F) \right\}
\]

\[
G^*_k(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + \min_{u_{k+1}, \ldots, u_K} \left\{ \sum_{i=k+1}^{K} l(x_i, u_i) + l_F(x_F) \right\} \right\}
\]

\[
G^*_k(x_k) = \min_{u_k} \left\{ l(x_k, u_k) + G^*_{k+1}(x_{k+1}) \right\}
\]
**Any length optimal planning**

\[ G^*(x) = \min_u \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

**Optimal cost to go**

\[ u^* = \arg \min_{u \in U(x)} \left\{ l(x, u) + G^*(f(x, u)) \right\} \]

**Optimal action**
Policy iteration

• Policy evaluation
• Policy improvement
Policy iteration

Policy evaluation

\[ \pi_0 \xrightarrow{E} V^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V^{\pi_1} \xrightarrow{I} \ldots \xrightarrow{I} \pi^* \xrightarrow{E} V^* \]

Policy improvement

“greedification”
Iterative policy evaluation

Estimate values of states following a certain policy

• Initialize $V_0(s) = 0$ for all states $s$ in $S$
• Update rule based on Bellman equation:

$$V_{k+1}(s) = E_{\pi} \left\{ r_{t+1} + \gamma V_k(s_{t+1}) \mid s_t = s \right\}$$

$$= \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right]$$

• Repeat until values converge for all states
• Output $V \approx V^\pi$
Policy improvement

Given two deterministic policies $\pi$ & $\pi'$:
If $Q^\pi(s, \pi'(s)) \geq V^\pi(s)$, for all $s$ in $S$
Then $V^{\pi'}(s) \geq V^\pi(s)$, for all $s$ in $S$

While following policy $\pi$, if changing one action according to $\pi'$ yields higher or equal return then $\pi'$ must be better or as good as $\pi$. 
Policy improvement

Find new greedy policy $\pi'$ given policy $\pi$:

$$\pi'(s) = \arg\max_a Q^\pi(s, a)$$

$$= \arg\max_a E \left\{ r_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s, a_t = a \right\}$$

$$= \arg\max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^\pi(s') \right]$$

If $V^\pi = V^{\pi'}$

$$V^{\pi'}(s) = \max_a E_{\pi'} \left\{ r_{t+1} + \gamma V^{\pi'}(s_{t+1}) \mid s_t = s, a_t = a \right\}$$

$$= \max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^{\pi'}(s') \right]$$

Policy improvement must result in a strictly better policy unless the original policy is already optimal.

Bellman optimality equation
Example: gridworld

- $R_t = -1$ on all transitions
- $\gamma = 1$ (no discount)
- State transition: deterministic (no fail).
- Actions that would result in outside the grid put the agent back to the original state.

[Sutton & Barto 2nd edition, Example 4.1]
Exercise: Convergence of iterative policy evaluation

\[ V_{k+1}(s) = E_{\pi} \left\{ r_{t+1} + \gamma V_k(s_{t+1}) \mid s_t = s \right\} \]

\[ = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right] \]

\[ \pi'(s) = \arg\max_a Q^\pi(s, a) \]

\[ = \arg\max_a E \left\{ r_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s, a_t = a \right\} \]

\[ = \arg\max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^\pi(s') \right] \]

\[ V_1(s_1) = \text{up}(0.25*(-1+0)) + \text{down}() + \text{left}() + \text{right}() = \text{________} \]

\[ \pi(s_1) = \arg\max_a [\text{up}(-1 + (-1)), \text{down}(), \text{left}(), \text{right}()] = \text{________} \]
Convergence of iterative policy evaluation

$V_k$ for random policy

Greedy policy w.r.t. $V_k$

$V_{k+1}(s) = E_{\pi} \left\{ r_{t+1} + \gamma V_k(s_{t+1}) \middle| s_t = s \right\}$

$= \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right]$ 

$\pi'(s) = \arg\max_a Q^\pi(s, a)$

$= \arg\max_a E \left\{ r_{t+1} + \gamma V^\pi(s_{t+1}) \middle| s_t = s, a_t = a \right\}$

$= \arg\max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^\pi(s') \right]$ 

$V_1(s_1) = \text{up(.25*(-1+0))} + \text{down(.25*(-1+0))} + \text{left(.25*(-1+0))} + \text{right(.25*(-1+0))} = -1$

$\pi(s_1) = \arg\max_a [\text{up(-1-1), down(-1-1), left(-1+0), right(-1,-1)] = left}$
Convergence of iterative policy evaluation

$V_k$ for random policy

Greedy policy w.r.t. $V_k$

Optimal policy

[Sutton & Barto 2nd edition, Fig. 4.1]

58
Policy iteration

Policy evaluation
\[ \pi_0 \xrightarrow{E} V^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V^{\pi_1} \xrightarrow{I} \ldots \xrightarrow{I} \pi^* \xrightarrow{E} V^* \]

Policy improvement
“greedification”
Policy iteration

1. Initialize for all \( s \) in \( S \): \( V(s) \in \mathbb{R}, \pi(s) = A(s) \)

2. Policy evaluation
   Repeat until values converge:
   \[
   V_{k+1}(s) = \sum_a \pi(a \mid s) \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right], \forall s \in S
   \]

3. Policy improvement:
   \[
   \pi'(s) = \operatorname{argmax}_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V^\pi(s') \right], \forall s \in S
   \]
   – If policy stays stable stop; else go to 2
Value iteration

- Use Bellman optimality equation as update (or backup) function

\[ V_{k+1}(s) = \max_a \sum_{s'} p(s' | s, a) \left[ E(r_{t+1} | s, a, s') + \gamma V_k(s') \right], \forall s \in S \]
Value iteration

• Initialize $V(s)=0$ for all $s$ in $S$

• Repeat until values converge:

\[ V_{k+1}(s) = \max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right], \forall s \in S \]

• Output policy

\[ \pi(s) = \arg\max_a \sum_{s'} p(s' \mid s, a) \left[ E(r_{t+1} \mid s, a, s') + \gamma V_k(s') \right], \forall s \in S \]
Summary

• Reinforcement learning and MDP formulation
• Dynamic programming for solving MDP
• Curse of dimensionality: number of states grow exponentially in the number of state variables
• Next
  – Exploration vs. exploitation
  – Reward shaping