

Uczenie ze wzmocnieniem



Piotr Duch

pduch@iis.p.lodz.pl
Instytut Informatyki Stosowanej
Politechnika Łódzka

Zima 2022

Plan wykładu

- 1 Wprowadzenie
- 2 Algorytmy w systemach wieloagentowych
- 3 Podstawowe pojęcia
- 4 Uczenie pasywne
- 5 Uczenie aktywne
- 6 Aproksymacja funkcji wartości stanu



Informacje ogólne:

- Materiały wykładowe oraz laboratoryjne dostępne są na stronie (pduch.iis.p.lodz.pl).
- Literatura podstawowa:
 - Richard S. Sutton, and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
 - Morales, Miguel. *Grokking deep reinforcement learning*. Simon and Schuster, 2020.
- Wykłady uzupełniające:
 - RL Course by David Silver - <https://www.youtube.com>
 - CS 188: Artificial Intelligence by Pieter Abbeel (wykład 10 i 11)- <https://www.youtube.com/watch?v=IXuHxkpO5E8>
- Materiały dodatkowe:
 - Practical RL Course by Yandex School of Data Analysis - https://github.com/yandexdataschool/Practical_RL
 - CS 188: Introduction to Artificial Intelligence by Berkeley University of California - <https://inst.eecs.berkeley.edu/cs188/fa19/project3/>



Informacje ogólne - zaliczenie:

■ Projekt:

- Zadania do wykonania w Pythonie (3 notebooki pythonowe, jeden projekt),
- Projekt własnej gry w Pythonie wraz z implementacją wybranych algorytmów.
- Rozbudowa projektu Pacman (implementacja algorytmu aproksymacji funkcji wartości stanu),
- Ocena końcowa:
 - Część I - 66% - 73% ocena 3, 73% - 80% ocena 3.5, 80% - 87% ocena 4, 87% - 94% ocena 4.5, 94% i więcej - 5.
 - Część II - ocena projektu.
 - Część III - ocena na podstawie turnieju botów.
 - Ocena końcowa jest oceną ważoną z każdej części (30%, 40%, 40%).
Konieczne jest uzyskanie pozytywnej oceny z każdej części.

■ Wykład - ??.

■ Kontakt:

- poprzez platformę MS Teams na chacie indywidualnym,
- mailowo: pduch@iis.p.lodz.pl.



Informacje szczegółowe - plan działania:

- Minimax, Alpha-Beta, Expectimax - *Project 2: Multi-Agent Search, Berkeley*.
- MCTS - *Monet Carlo Tree Search* - implementacja we własnym projekcie.
- Uczenie pasywne (*Policy Evaluation, Policy Improvement, Policy Iteration, Value Iteration*) - Notebook Pythonowy 1 + implementacja wybranego algorytmu we własnym projekcie (Pliki dodatkowe do notebooków).
- Uczenie aktywne (*Q-Learning, Sarsa, Expected Sarsa, Sarsa (λ), Double Q-Learning*) - Notebook Pythonowy 2 i Notebook Pythonowy 3 + implementacja wybranego algorytmu we własnym projekcie.
- Aproksymacja funkcji wartości - implementacja we własnym projekcie + implementacja w pacmanie (link zostanie dodany później).



Uczenie ze wzmocnieniem

Wprowadzenie



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Uczenie maszynowe:



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Uczenie maszynowe:

- Uczenie z nadzorem:
 - Klasyfikacja.
 - Regresja (predykcja).



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Uczenie maszynowe:

- Uczenie z nadzorem:
 - Klasyfikacja.
 - Regresja (predykcja).
- Uczenie bez nadzoru:
 - Grupowanie (m.in. klasteryzacja, analiza skupień).
 - Redukcja wymiarów.
 - Uzupełnianie wartości.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Uczenie maszynowe:

- Uczenie z nadzorem:
 - Klasyfikacja.
 - Regresja (predykcja).
- Uczenie bez nadzoru:
 - Grupowanie (m.in. klasteryzacja, analiza skupień).
 - Redukcja wymiarów.
 - Uzupełnianie wartości.
- Uczenie ze wzmocnieniem.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co odróżnia uczenie ze wzmocnieniem od innych działów uczenia maszynowego:



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co odróżnia uczenie ze wzmocnieniem od innych działów uczenia maszynowego:

- Nie potrzebna jest baza danych - agent uczy się na podstawie informacji o nagrodach otrzymywanych ze środowiska.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co odróżnia uczenie ze wzmocnieniem od innych działów uczenia maszynowego:

- Nie potrzebna jest baza danych - agent uczy się na podstawie informacji o nagrodach otrzymywanych ze środowiska.
- Nagroda może być odłożona w czasie.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co odróżnia uczenie ze wzmocnieniem od innych działów uczenia maszynowego:

- Nie potrzebna jest baza danych - agent uczy się na podstawie informacji o nagrodach otrzymywanych ze środowiska.
- Nagroda może być odłożona w czasie.
- Czas ma znaczenie.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co odróżnia uczenie ze wzmocnieniem od innych działów uczenia maszynowego:

- Nie potrzebna jest baza danych - agent uczy się na podstawie informacji o nagrodach otrzymywanych ze środowiska.
- Nagroda może być odłożona w czasie.
- Czas ma znaczenie.
- Działanie agenta ma wpływ na dane, jakie otrzymuje ze środowiska.



Uczenie ze wzmocnieniem - wprowadzenie

Wprowadzenie

Co to jest:

- **Agent** wchodzi w interakcję ze **środowiskiem**, w którym chce osiągnąć określony **cel**.
- **Akcje** podjęte przez agenta są oceniane przez środowisko.
- W wyniku wykonania wybranej akcji, agent otrzymuje **nagrodę** (może być pozytywna lub negatywna).



Uczenie ze wzmocnieniem - wprowadzenie

Metody

- Uczenie pasywne:
 - Ocena strategii (ang. *Policy Evaluation*)
 - Polepszanie strategii (ang. *Policy Improvement*)
 - Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)
 - Iteracyjne obliczanie funkcji wartości (ang. *Value Iteration*)



Uczenie ze wzmocnieniem - wprowadzenie

Metody cd.

- Uczenie aktywne:
 - Metody różnic czasowych (ang. *Temporal Difference Learning*)
 - Monte Carlo
 - Q-Learning
 - SARSA
 - Metody aproksymacyjne
 - Aproksymacja funkcji wartości (ang. *Approximate Q-Learning*)
 - Metody wykorzystujące głębokie sieci neuronowe
 - Deep Q-Learning
 - Double Q-Learning
 - Actor-Critic
 - REINFORCE
 - Policy Gradient



Wprowadzenie

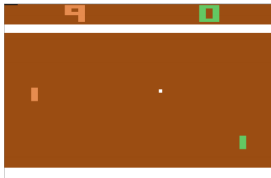
Przykłady



(a) Space Invaders



(b) Beam Rider



(c) Pong



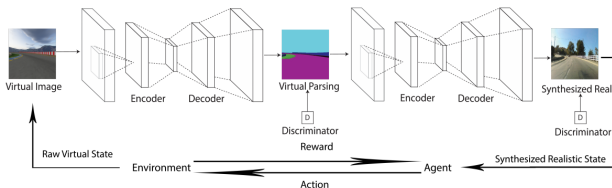
(d) Breakout

Rysunek 1: Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." (2013).



Wprowadzenie

Przykłady cd.



Pan, Xinlei, et al. "Virtual to real reinforcement learning for autonomous driving." arXiv preprint arXiv:1704.03952 (2017).



Wprowadzenie

Przykłady cd.

A: Where are you going? (1)

B: I'm going to the police station. (2)

A: I'll come with you. (3)

B: No, no, no, no, you're not going anywhere. (4)

A: Why? (5)

B: I need you to stay here. (6)

A: I don't know what you are talking about. (7)

A: How old are you? (1)

B: I'm 16. Why are you asking? (2)

A I thought you were 12. (3)

B: What made you think so? (4)

A: I don't know what you are talking about. (5)

B: You don't know what you are saying. (6)

Li, Jiwei, et al. "Deep reinforcement learning for dialogue generation." (2016).



Wprowadzenie

Przykłady cd.

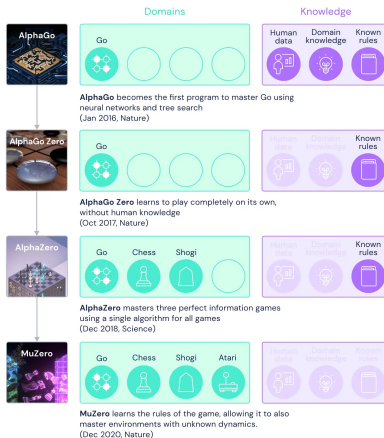
Obrót akcjami:

- Dempster, Michael AH, and Vasco Leemans. "An automated FX trading system using adaptive reinforcement learning." *Expert Systems with Applications* 30.3 (2006): 543-552.
- Xiong, Zhuoran, et al. "Practical deep reinforcement learning approach for stock trading." *arXiv preprint arXiv:1811.07522* (2018).
- Carapuço, João, Rui Neves, and Nuno Horta. "Reinforcement learning applied to Forex trading." *Applied Soft Computing* 73 (2018): 783-794.



Wprowadzenie

Przykłady cd.



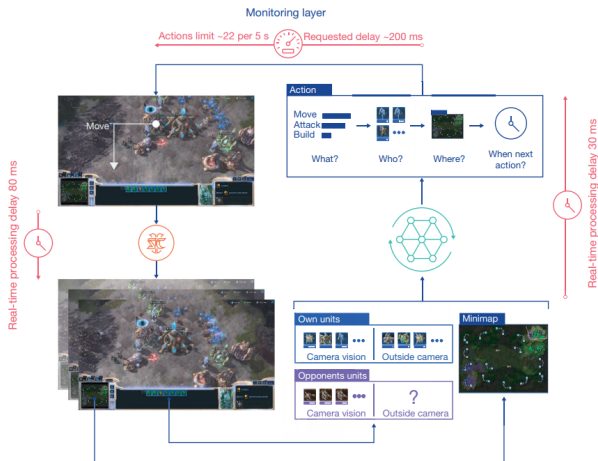
Rysunek 3: Od AlphaGo do MuZero



Wprowadzenie

Przykłady cd.

a



Rysunek 4: AlphaStar



Algorytmy w systemach wieloagentowych (ang. *Multi-agent search algorithms*)



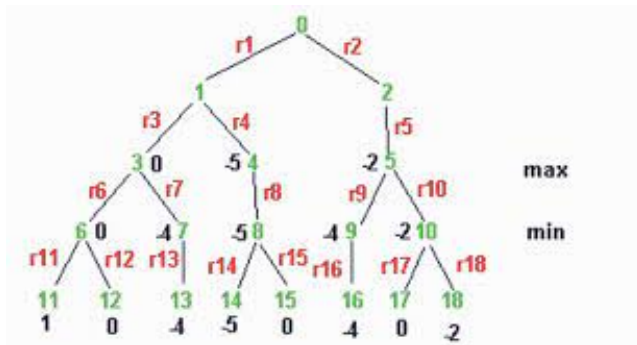
Algorytmy

- Minimax
- Alpha-Beta
- Monte Carlo Tree Search (MCTS)



Algorytmy

Minimax

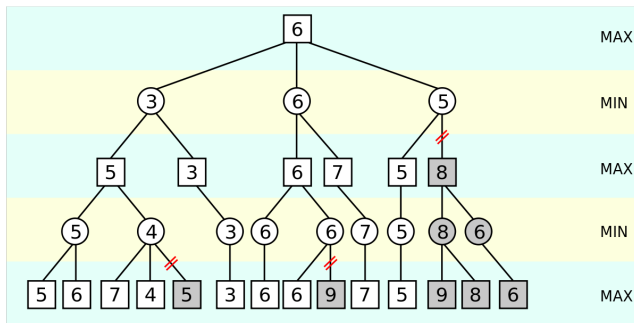


Rysunek 5: Minimax



Algorytmy

Alpha-Beta

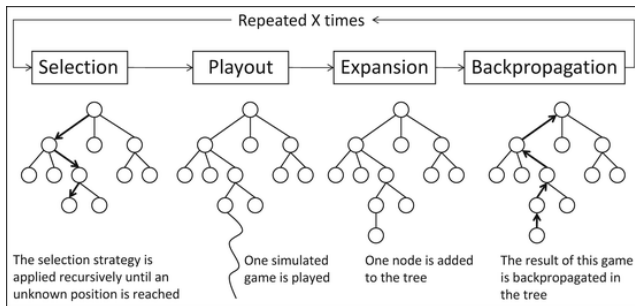


Rysunek 6: Alpha-Beta - Wikipedia



Algorytmy

MCTS



Rysunek 7: Monte-Carlo Tree Search in Board Games



Algorytmy

MCTS

- Selekcja - wybieramy ścieżkę od początkowego wężła do najbardziej obiecującego liści.



Algorytmy

MCTS

- Selekcja - wybieramy ścieżkę od początkowego węzła do najbardziej obiecującego liścia.

$$UCB(node_i) = \frac{w_i}{n_i} + c \sqrt{\frac{\log N}{n_i}} \quad (1)$$

- Ekspansja - rozwinięcie, wybieramy losowy węzeł z ostatniego liścia.
- Symulacja (Roll-out) - rozgrywamy wiele gier losowo zapamiętując wyniki.
- Propagacja wsteczna - aktualizujemy wartości wcześniejszych węzłów.



Algorytmy

Projekt

- Minimax, Alpha-Beta, Expectimax - *Project 2: Multi-Agent Search, Berkeley.*
- MCTS - implementacja we własnym projekcie.



Uczenie ze wzmocnieniem

Podstawowe pojęcia



Podstawowe pojęcia

Interakcja agent - środowisko



Podstawowe pojęcia

Środowisko (ang. *Environment*, np. plansza do gry Pacman):

- Opisuje świat, z którym agent wchodzi w interakcję
- Wejście:
 - Akcja
- Wyjście:
 - Stan
 - Nagroda



Podstawowe pojęcia

Agent (ang. *Agent*):

- Poprzez interakcję ze środowiskiem uczy się, jak osiągnąć założony cel
- Wejście:
 - Stan
 - Nagroda
- Wyjście:
 - Akcja



Podstawowe pojęcia

Nagroda (ang. *Reward*):

- Wartość zwracana przez środowisko w momencie wykonania akcji wybranej przez agenta.
- Reprezentuje cel, lub cele, jakie agent ma osiągnąć.
- Oznaczenie: r_t - nagroda otrzymana w chwili czasu t .



Podstawowe pojęcia

Oczekiwana nagroda (ang. *Return*):

- Oczekiwana nagroda po zakończeniu bieżącego epizodu.
- Celem uczenia za wzmocnieniem jest maksymalizacja nagrody oczekiwanej.
- Oznaczenie: G_t - oczekiwana nagroda w chwili czasu t .

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T \quad (2)$$



Podstawowe pojęcia

Akcja (ang. *Action*):

- Dyskretna (1 z N dostępnych w danym środowisku).
- Ciągła (wartość lub wektor wartości).



Podstawowe pojęcia

Strategia (ang. *Policy*):

- Zasady, według których wybierana jest akcja w danym stanie.
- Może być deterministyczna lub stochastyczna.
- Oznaczenie: π .



Podstawowe pojęcia

Funkcja wartości (oceny, ang. *Value function*):

- Określa, jak dobrze być w danym stanie.
- Oczekiwana suma nagród, jaką możemy otrzymać, rozpoczynając w stanie s i działając zgodnie ze strategią π .
- Rodzaje:
 - Funkcja wartości stanu $V^\pi(s)$ - jak dobrze być w stanie s , działając zgodnie ze strategią π .
 - Funkcja wartości stanu-akcji $Q^\pi(s, a)$ - jak dobrze będąc w stanie s jest wykonać akcję a , działając zgodnie ze strategią π .



Podstawowe pojęcia

Stan (ang. *State*):

- Zbiór wartości opisujących aktualną sytuację.
- Jest podstawą wyboru akcji przez agenta zgodnie z jego strategią.



Podstawowe pojęcia

Proces decyzyjny Markowa

Własność Markowa

Własność procesów stochastycznych polegająca na tym, że warunkowe rozkłady prawdopodobieństwa przyszłych stanów procesu są zdeterminowane wyłącznie przez jego bieżący stan, bez względu na przeszłość.

Wikipedia



Podstawowe pojęcia

Proces decyzyjny Markowa

Proces decyzyjny Markowa (ang. *Markov Decision Process* (MDP))

Ciąg zdarzeń, w którym prawdopodobieństwo każdego zdarzenia zależy jedynie od wyniku poprzedniego. W ujęciu matematycznym, procesy Markowa to takie procesy stochastyczne, które spełniają własność Markowa.

Wikipedia



Podstawowe pojęcia

Proces decyzyjny Markowa

Elementy MDP:



Podstawowe pojęcia

Proces decyzyjny Markowa

Elementy MDP:

- Stany:
 - S - zbiór wszystkich możliwych stanów.
 - s - pojedynczy stan ($s \in S$).
 - s_0 - stan początkowy ($s_0 \in S$).



Podstawowe pojęcia

Proces decyzyjny Markowa

Elementy MDP:

■ Stany:

- S - zbiór wszystkich możliwych stanów.
- s - pojedynczy stan ($s \in S$).
- s_0 - stan początkowy ($s_0 \in S$).

■ Akcje:

- A - zbiór wszystkich możliwych akcji.
- a - pojedyncza akcja.
- $A(s)$ - zbiór akcji możliwych do wykonania w stanie s .



Podstawowe pojęcia

Proces decyzyjny Markowa

Elementy MDP:

■ Stany:

- S - zbiór wszystkich możliwych stanów.
- s - pojedynczy stan ($s \in S$).
- s_0 - stan początkowy ($s_0 \in S$).

■ Akcje:

- A - zbiór wszystkich możliwych akcji.
- a - pojedyncza akcja.
- $A(s)$ - zbiór akcji możliwych do wykonania w stanie s .

■ Model środowiska:

- $P(s'|s, a)$ - prawdopodobieństwo przejścia ze stanu s do stanu s' , wykonując akcję a .
- $P(s', r|s, a)$ - prawdopodobieństwo przejścia ze stanu s do stanu s' i otrzymania nagrody r , wykonując akcję a .



Podstawowe pojęcia

Proces decyzyjny Markowa

Elementy MDP:

- Stany:
 - S - zbiór wszystkich możliwych stanów.
 - s - pojedynczy stan ($s \in S$).
 - s_0 - stan początkowy ($s_0 \in S$).
- Akcje:
 - A - zbiór wszystkich możliwych akcji.
 - a - pojedyncza akcja.
 - $A(s)$ - zbiór akcji możliwych do wykonania w stanie s .
- Model środowiska:
 - $P(s'|s, a)$ - prawdopodobieństwo przejścia ze stanu s do stanu s' , wykonując akcję a .
 - $P(s', r|s, a)$ - prawdopodobieństwo przejścia ze stanu s do stanu s' i otrzymania nagrody r , wykonując akcję a .
- Funkcja nagrody $R(s)$.



Podstawowe pojęcia

Proces decyzyjny Markowa



Podstawowe pojęcia

Funkcje wartości

- Funkcja wartości stanu dla strategii π .

$$v_{\pi}(s) \doteq \mathbb{E}[G_t | S_t = s] \quad (3)$$



Podstawowe pojęcia

Funkcje wartości

- Funkcja wartości stanu dla strategii π .

$$v_{\pi}(s) \doteq \mathbb{E}[G_t | S_t = s] \quad (3)$$

- Funkcja wartości stanu-akcji dla strategii π .

$$q_{\pi}(s, a) \doteq \mathbb{E}[G_t | S_t = s, A_t = a] \quad (4)$$



Podstawowe pojęcia

Funkcje wartości

- Funkcja wartości stanu dla strategii π .

$$v_{\pi}(s) \doteq \mathbb{E}[G_t | S_t = s] \quad (3)$$

- Funkcja wartości stanu-akcji dla strategii π .

$$q_{\pi}(s, a) \doteq \mathbb{E}[G_t | S_t = s, A_t = a] \quad (4)$$

- Funkcje wartości mogą być obliczane na podstawie doświadczenia.



Podstawowe pojęcia

Równanie Bellmana

$$\begin{aligned}v_{\pi}(s) &\doteq \mathbb{E}[G_t | S_t = s] \\&= \mathbb{E}[R_{t+1} + \gamma G_{t+1} | S_t = s] \\&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma \mathbb{E}[G_{t+1} | S_{t+1} = s]] \\&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma v_{\pi}(s')], \text{ dla wszystkich } s \in S.\end{aligned}$$

(5)



Podstawowe pojęcia

Równanie Bellmana

$$v_{\pi}(s) \doteq \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_{\pi}(s')]$$



Podstawowe pojęcia

Równanie Bellmana

Prawdopodobieństwo wyboru

akcji a w stanie s zgodnie

z założoną strategią π

$$v_{\pi}(s) \doteq \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_{\pi}(s')]$$



Podstawowe pojęcia

Równanie Bellmana

Prawdopodobieństwo wyboru

akcji a w stanie s zgodnie

z założoną strategią π

$$v_{\pi}(s) \doteq \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_{\pi}(s')]$$

Prawdopodobieństwo przejścia

do stanu s' , i otrzymania

nagrody r , ze stanu s

wybierając akcję a zgodnie

z założoną strategią π



Podstawowe pojęcia

Równanie Bellmana

Prawdopodobieństwo wyboru
akcji a w stanie s zgodnie
z założoną strategią π

$$v_{\pi}(s) \doteq \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_{\pi}(s')]$$

Prawdopodobieństwo przejścia
do stanu s' , i otrzymania
nagrody r , ze stanu s
wybierając akcję a zgodnie
z założoną strategią π

Nagroda otrzymana w
wyniku wykonania akcji a
oraz przejścia do stanu s'



Podstawowe pojęcia

Równanie Bellmana

Prawdopodobieństwo wyboru
akcji a w stanie s zgodnie
z założoną strategią π

$$v_{\pi}(s) \doteq \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_{\pi}(s')]$$

Prawdopodobieństwo przejścia
do stanu s' , i otrzymania
nagrody r , ze stanu s
wybierając akcję a zgodnie
z założoną strategią π

Nagroda otrzymana w
wyniku wykonania akcji a
oraz przejścia do stanu s'

Przewidywana skumulowana
nagroda w stanie s'



Podstawowe pojęcia

Materiały uzupełniające

- Książka *Reinforcement Learning: An Introduction*, Richard S. Sutton and Andrew G. Barto, wydanie drugie, 2018.
 - Rozdział 1 - *Introduction*.
 - Rozdział 3 - *Finite Markov Decision Processes*.
- Książka *Grokking Deep Reinforcement Learning*, Miguel Morales, 2020.
 - Rozdział 2 - *Mathematical foundations of reinforcement learning*.
- Video *RL Course by David Silver - Lecture 1: Introduction to Reinforcement Learning*.



Uczenie pasywne

(ang. *model based learning*)



Uczenie pasywne

Algorytmy:

- Ocena strategii (ang. *Policy Evaluation*).
- Polepszanie strategii (ang. *Policy Improvement*).
- Iteracyjne doskonalenie strategii (ang. *Policy Iteration*).
- Iteracyjne obliczanie funkcji wartości (ang. *Value Iteration*).



Uczenie pasywne

Ocena strategii



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*)

Cele:

- Oszacowanie wartości dla każdego stanu, dla założonej strategii.
- Wyznaczenie $v_{\pi}(s)$ dla każdego $s \in S$, dla założonej strategii π .



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*)

Cele:

- Oszacowanie wartości dla każdego stanu, dla założonej strategii.
- Wyznaczenie $v_\pi(s)$ dla każdego $s \in S$, dla założonej strategii π .

Rozwiązanie:

- Metoda iteracyjna:

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma V_k(s')]$$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*)

Cele:

- Oszacowanie wartości dla każdego stanu, dla założonej strategii.
- Wyznaczenie $v_\pi(s)$ dla każdego $s \in S$, dla założonej strategii π .

Rozwiązanie:

- Metoda iteracyjna:

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma V_k(s')]$$

Warianty:

- Dwie macierze.
- Obliczenia w miejscu.



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*)

Ocena strategii

Wejście:

- π - strategia, która ma zostać oszacowana.
- θ - dokładność szacowania strategii.

Wyjście:

- $V(s)$ - funkcja wartości stanów wyznaczona dla strategii π .

Inicjalizacja tablicy wartości stanów $V(s)$ losowymi wartościami, za wyjątkiem stanu końcowego, któremu przypisana jest wartość 0.

Licz:

$$\Delta \leftarrow 0$$

Dla każdego $s \in S$:

$$v \leftarrow V(s)$$

$$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_\pi(s')]$$

$$\Delta \leftarrow \max(\Delta, |v - V(s)|)$$

Dopóki $\Delta > \theta$

Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*)

Iterative Policy Evaluation

Input π , the policy to be evaluated

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation

Initialize $V(s)$, for all $s \in S^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in S$:

$v \leftarrow V(s)$

$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma v_\pi(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

until $\Delta > \theta$

Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy

Model środowiska:

Strategia:

W każdym stanie prawdopodobieństwo wyboru każdej z możliwych akcji jest takie samo.



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Wejście:

- Stany:

- $S = \{s_0, s_1, s_2\}$.

- Akcje:

- $A_{s_0} = \{a_0, a_1\}$.

- $A_{s_1} = \{a_0, a_1\}$.

- $A_{s_2} = \{a_0, a_1\}$.

- Funkcja wartości:

- $V(s_0) = 0, V(s_1) = 0, V(s_2) = 0$.

- Strategia:

- $\pi(a_0|s_0) = \pi(a_1|s_0) = 0.5$.

- $\pi(a_0|s_1) = \pi(a_1|s_1) = 0.5$.

- $\pi(a_0|s_2) = \pi(a_1|s_2) = 0.5$.



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Wejście:

- Prawdopodobieństwo przejścia:

- $p(s_2|s_0, a_0) = 0.5$, $p(s_0|s_0, a_0) = 0.5$, $p(s_2|s_0, a_1) = 1$.
- $p(s_0|s_1, a_0) = 0.7$, $p(s_1|s_1, a_0) = 0.1$, $p(s_2|s_1, a_0) = 0.2$,
 $p(s_1|s_1, a_1) = 0.95$, $p(s_2|s_1, a_1) = 0.05$.
- $p(s_0|s_2, a_0) = 0.4$, $p(s_2|s_2, a_0) = 0.6$, $p(s_0|s_2, a_1) = 0.3$,
 $p(s_1|s_2, a_1) = 0.3$, $p(s_2|s_2, a_1) = 0.4$.

- Nagroda:

- $r(s_1, a_0, s_0) = 5$.
- $r(s_2, a_1, s_0) = -1$.
- W pozostałych przypadkach: $r = 0$.

- Discount factor:

- $\gamma = 0.9$.



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Dla każdego $s \in S$:

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma V_k(s')]$$

Krok 1 (dla stanu s_0):

Potrzebne dane:

- $\pi(a_0|s_0) = \pi(a_1|s_0) = 0.5$.
- $p(s_2|s_0, a_0) = 0.5$, $p(s_0|s_0, a_0) = 0.5$, $p(s_2|s_0, a_1) = 1$.
- $r = 0$.
- $V_0(s_0) = V_0(s_1) = V_0(s_2) = 0$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Krok 1 (dla stanu s_0):

$$V_1(s_0) = \pi(a_0|s_0) * [p(s_2|s_0, a_0) * (r + \gamma * V_0(s_2)) + \\ p(s_0|s_0, a_0) * (r + \gamma * V_0(s_0))] + \\ \pi(a_1|s_0) * [p(s_2|s_0, a_1) * (r + \gamma * V_0(s_2))]$$

Po podstawieniu wartości otrzymujemy:

$$V_1(s_0) = 0.5 * [0.5 * (0 + 0.9 * 0) + \\ 0.5 * (0 + 0.9 * 0)] + \\ 0.5 * [1 * (0 + 0.9 * 0)]$$

Ostatecznie otrzymujemy:

$$V_1(s_0) = 0$$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Dla każdego $s \in S$:

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma V_k(s')]$$

Krok 1 (dla stanu s_1):

Potrzebne dane:

- $\pi(a_0|s_1) = \pi(a_1|s_1) = 0.5$.
- $p(s_0|s_1, a_0) = 0.7$, $p(s_1|s_1, a_0) = 0.1$, $p(s_2|s_1, a_0) = 0.2$,
 $p(s_1|s_1, a_1) = 0.95$, $p(s_2|s_1, a_1) = 0.05$.
- $r(s_1, a_0, s_0) = 5$, $r = 0$.
- $V_0(s_0) = V_0(s_1) = V_0(s_2) = 0$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Krok 1 (dla stanu s_1):

$$\begin{aligned}
 V_1(s_1) = & \pi(a_0|s_1) * [p(s_0|s_1, a_0) * (r(s_1, a_0, s_0) + \gamma * V_0(s_0)) + \\
 & p(s_1|s_1, a_0) * (r + \gamma * V_0(s_1)) + \\
 & p(s_2|s_1, a_0) * (r + \gamma * V_0(s_2))] + \\
 & \pi(a_1|s_1) * [p(s_1|s_1, a_1) * (r + \gamma * V_0(s_1)) + \\
 & p(s_2|s_1, a_1) * (r + \gamma * V_0(s_2))]
 \end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}
 V_1(s_1) = & 0.5 * [0.7 * (5 + 0.9 * 0) + \\
 & 0.1 * (0 + 0.9 * 0) + \\
 & 0.2 * (0 + 0.9 * 0)] + \\
 & 0.5 * [0.95 * (0 + 0.9 * 0) + \\
 & 0.05 * (0 + 0.9 * 0)]
 \end{aligned}$$

Ostatecznie otrzymujemy:

$$V_1(s_1) = 1.75$$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Dla każdego $s \in S$:

$$V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) [r + \gamma V_k(s')]$$

Krok 1 (dla stanu s_2):

Potrzebne dane:

- $\pi(a_0|s_2) = \pi(a_1|s_2) = 0.5$.
- $p(s_0|s_2, a_0) = 0.4$, $p(s_2|s_2, a_0) = 0.6$, $p(s_0|s_2, a_1) = 0.3$,
 $p(s_1|s_2, a_1) = 0.3$, $p(s_2|s_2, a_1) = 0.4$.
- $r(s_2, a_1, s_0) = -1$, $r = 0$.
- $V_0(s_0) = V_0(s_1) = V_0(s_2) = 0$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Krok 1 (dla stanu s_2):

$$\begin{aligned}
 V_1(s_2) = & \pi(a_0|s_2) * [p(s_0|s_2, a_0) * (r + \gamma * V_0(s_0)) + \\
 & p(s_2|s_2, a_0) * (r + \gamma * V_0(s_2))] + \\
 & \pi(a_1|s_2) * [p(s_0|s_2, a_1) * (r(s_2, a_1, s_0) + \gamma * V_0(s_0)) + \\
 & p(s_1|s_2, a_1) * (r + \gamma * V_0(s_1)) + \\
 & p(s_2|s_2, a_1) * (r + \gamma * V_0(s_2))]
 \end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}
 V_1(s_2) = & 0.5 * [0.4 * (0 + 0.9 * 0) + \\
 & 0.6 * (0 + 0.9 * 0)] + \\
 & 0.5 * [0.3 * (-1 + 0.9 * 0) + \\
 & 0.4 * (0 + 0.9 * 0) + \\
 & 0.3 * (0 + 0.9 * 0)]
 \end{aligned}$$

Ostatecznie otrzymujemy:

$$V_1(s_2) = -0.15$$



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Wartości funkcji dla poszczególnych stanów wyznaczone po n krokach algorytmu szacowania strategii.

Stan	Start	Krok 1	Krok 5	Krok 25	Krok 73
$V(s_0)$	0	0	0.24	1.32	1.47
$V(s_1)$	0	1.75	3.31	4.40	4.55
$V(s_2)$	0	-0.15	0.46	1.54	1.69



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - przykład liczbowy cd.

Wartości funkcji dla poszczególnych stanów wyznaczone po n krokach algorytmu szacowania strategii - implementacja *in-place*.

Stan	Start	Krok 1	Krok 5	Krok 25	Krok 53
$V(s_0)$	0	0	0.49	1.42	1.47
$V(s_1)$	0	1.75	3.56	4.51	4.55
$V(s_2)$	0	0.09	0.76	1.64	1.69



Uczenie pasywne

Ocena strategii (ang. *Policy Evaluation*) - ćwiczenie

Implementacja algorytmu oceny strategii w pliku *PolicyEvaluation.py*:

- implementacja algorytmu z wykorzystaniem dwóch osobnych macierzy do obliczeń (funkcja *policy_eval_two_arrays*),
- implementacja algorytmu z wykorzystaniem obliczeń w miejscu (funkcja *policy_eval_in_place*).



Uczenie pasywne

Poprawa strategii



Uczenie pasywne

Poprawa strategii (ang. *Policy Improvement*)

Problem:

- Jak można poprawić aktualną strategię?

Twierdzenie o poprawie strategii (ang. *Policy Improvement Theorem*)

$$q_{\pi}(s, a) \doteq \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi}(s')] \quad (6)$$

$$\forall s \in S, q_{\pi}(s, \pi'(s)) \geq v_{\pi}(s) \rightarrow v'_{\pi}(s) \geq v_{\pi}(s) \quad (7)$$



Uczenie pasywne

Poprawa strategii (ang. *Policy Improvement*)

Zgodnie z twierdzeniem o poprawie strategii, zastosowanie zachłannej strategii zawsze będzie lepsze, bądź równe obecnej strategii:

$$\pi'(s) = \operatorname{argmax}_a \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma v_{\pi}(s')] \quad (8)$$



Uczenie pasywne

Iteracyjne doskonalenie strategii



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Cel:

- Określenie optymalnej strategii.



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Cel:

- Określenie optymalnej strategii.

Rozwiązanie:

- Oszacowanie aktualnej strategii (algorytm *Policy Evaluation*).
- Poprawienie aktualnej strategii (algorytm *Policy Improvement*).



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Cel:

- Określenie optymalnej strategii.

Rozwiązanie:

- Oszacowanie aktualnej strategii (algorytm *Policy Evaluation*).
- Poprawienie aktualnej strategii (algorytm *Policy Improvement*).

Optymalna strategia:

- Jeżeli zastosowanie algorytmu *Policy Improvement* na aktualnej strategii jej nie zmieni, to oznacza, że aktualna strategia jest optymalna.



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Wyjście:

- $V \approx v_*$ - optymalna funkcja wartości stanów.
- $\pi \approx \pi_*$ - optymalna strategia.

Inicjalizacja tablicy strategii losowymi akcjami $\pi(s) \in A(s)$ dla $s \in S$.

Licz:

strategia_stabilna \leftarrow *true*

$V \leftarrow \text{ocena_strategii}(\pi)$

Dla każdego $s \in S$:

poprzednia_akcja $\leftarrow \pi(s)$

$\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma V(s')]$

Jeżeli *poprzednia_akcja* $\neq \pi(s)$ ustaw *strategia_stabilna* \leftarrow *false*

Dopóki *strategia_stabilna* \neq *true*



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*)

Policy Iteration

1. Initialization

$V(s) \in \mathbb{R}$ and $\pi(s) \in \mathbf{A}(s)$ arbitrarily for all $s \in S$.

2. Policy Evaluation

3. Policy Improvement

Licz:

$policy_stable \leftarrow true$

For each $s \in S$:

$old_action \leftarrow \pi(s)$

$\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s'} p(s', r|s, a)[r + \gamma V(s')]$

If $old_action \neq \pi(s)$, then $policy_stable \leftarrow false$

If $policy_stable == false$, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy

Model środowiska:

Strategia:

W każdym stanie wybieramy akcję a_0 .



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Wejście:

- Stany:

- $S = \{s_0, s_1, s_2\}$.

- Akcje:

- $A_{s_0} = \{a_0, a_1\}$.

- $A_{s_1} = \{a_0, a_1\}$.

- $A_{s_2} = \{a_0, a_1\}$.

- Strategia:

- $\pi(s_0) = a_0$.

- $\pi(s_1) = a_0$.

- $\pi(s_2) = a_0$.



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Wejście:

- Prawdopodobieństwo przejścia:

- $p(s_2|s_0, a_0) = 0.5$, $p(s_0|s_0, a_0) = 0.5$, $p(s_2|s_0, a_1) = 1$.
- $p(s_0|s_1, a_0) = 0.7$, $p(s_1|s_1, a_0) = 0.1$, $p(s_2|s_1, a_0) = 0.2$,
 $p(s_1|s_1, a_1) = 0.95$, $p(s_2|s_1, a_1) = 0.05$.
- $p(s_0|s_2, a_0) = 0.4$, $p(s_2|s_2, a_0) = 0.6$, $p(s_0|s_2, a_1) = 0.3$,
 $p(s_1|s_2, a_1) = 0.3$, $p(s_2|s_2, a_1) = 0.4$.

- Nagroda:

- $r(s_1, a_0, s_0) = 5$.
- $r(s_2, a_1, s_0) = -1$.
- W pozostałych przypadkach: $r = 0$.

- Discount factor:

- $\gamma = 0.9$.



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Funkcja wartości obliczona dla strategii π_0 (za pomocą algorytmu *Policy Evaluation*):

- $v_{\pi_0}(s_0) = 0.$
- $v_{\pi_0}(s_1) = 3.87.$
- $v_{\pi_0}(s_2) = 0.$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_0 :

$$\pi_1(s_0) = \operatorname{argmax}_a \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi_0}(s')]$$

Potrzebne dane:

- $\pi(s_0) = a_0$.
- $p(s_2 | s_0, a_0) = 0.5$, $p(s_0 | s_0, a_0) = 0.5$, $p(s_2 | s_0, a_1) = 1$.
- $r = 0$.
- $v_{\pi_0}(s_0) = 0.0$, $v_{\pi_0}(s_1) = 3.87$, $v_{\pi_0}(s_2) = 0.0$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_0 :

$$\begin{aligned}\pi_1(s_0) = \operatorname{argmax}_a (& p(s_2|s_0, a_0)[r + \gamma v_{\pi_0}(s_2)] + \\ & p(s_0|s_0, a_0)[r + \gamma v_{\pi_0}(s_0)], \\ & p(s_2|s_0, a_1)[r + \gamma v_{\pi_0}(s_2)])\end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}\pi_1(s_0) = \operatorname{argmax}_a (& 0.5[0 + 0.9 * 0] + \\ & 0.5[0 + 0.9 * 0], \\ & 1[0 + 0.9 * 0]) \\ = \operatorname{argmax}_a (& 0, 0)\end{aligned}$$

Ostatecznie nową akcją wybraną dla stanu s_0 jest akcja a_0 , czyli:

$$\pi_1(s_0) = a_0$$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_1 :

$$\pi_1(s_0) = \operatorname{argmax}_a \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi_0}(s')]$$

Potrzebne dane:

- $\pi(s_1) = a_0$.
- $p(s_0 | s_1, a_0) = 0.7$, $p(s_1 | s_1, a_0) = 0.1$, $p(s_2 | s_1, a_0) = 0.2$,
 $p(s_1 | s_1, a_1) = 0.95$, $p(s_2 | s_1, a_1) = 0.05$.
- $r(s_1, a_0, s_0) = 5$, $r = 0$.
- $v_{\pi_0}(s_0) = 0.0$, $v_{\pi_0}(s_1) = 3.87$, $v_{\pi_0}(s_2) = 0.0$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_1 :

$$\begin{aligned}\pi_1(s_1) = \operatorname{argmax}_a (& p(s_0|s_1, a_0)[r + \gamma v_{\pi_0}(s_0)] + \\ & p(s_1|s_1, a_0)[r + \gamma v_{\pi_0}(s_1)] + \\ & p(s_2|s_1, a_0)[r + \gamma v_{\pi_0}(s_2)], \\ & p(s_1|s_1, a_1)[r + \gamma v_{\pi_0}(s_1)] + \\ & p(s_2|s_1, a_1)[r + \gamma v_{\pi_0}(s_2)])\end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}\pi_1(s_1) = \operatorname{argmax}_a (& 0.7[5 + 0.9 * 0] + \\ & 0.1[0 + 0.9 * 3.87] + \\ & 0.2[0 + 0.9 * 0], \\ & 0.95[0 + 0.9 * 3.87] + \\ & 0.05[0 + 0.9 * 0]) \\ = \operatorname{argmax}_a (& 3.85, 3.31)\end{aligned}$$

Ostatecznie nową akcją wybraną dla stanu s_1 jest akcja a_0 , czyli:

$$\pi_1(s_1) = a_0$$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_2 :

$$\pi_1(s_0) = \operatorname{argmax}_a \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi_0}(s')]$$

Potrzebne dane:

- $\pi(s_2) = a_0$.
- $p(s_0 | s_2, a_0) = 0.4$, $p(s_2 | s_2, a_0) = 0.6$, $p(s_0 | s_2, a_1) = 0.3$,
 $p(s_1 | s_2, a_1) = 0.3$, $p(s_2 | s_2, a_1) = 0.4$.
- $r(s_2, a_1, s_0) = -1$, $r = 0$.
- $v_{\pi_0}(s_0) = 0.0$, $v_{\pi_0}(s_1) = 3.87$, $v_{\pi_0}(s_2) = 0.0$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_2 :

$$\begin{aligned}\pi_1(s_2) = \operatorname{argmax}_a (& p(s_0|s_2, a_0)[r + \gamma v_{\pi_0}(s_0)] + \\ & p(s_2|s_2, a_0)[r + \gamma v_{\pi_0}(s_2)], \\ & p(s_0|s_2, a_1)[r + \gamma v_{\pi_0}(s_0)] + \\ & p(s_1|s_2, a_1)[r + \gamma v_{\pi_0}(s_1)] + \\ & p(s_2|s_2, a_1)[r + \gamma v_{\pi_0}(s_2)])\end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}\pi_1(s_2) = \operatorname{argmax}_a (& 0.4[0 + 0.9 * 0] + \\ & 0.6[0 + 0.9 * 0], \\ & 0.3[-1 + 0.9 * 0] + \\ & 0.3[0 + 0.9 * 3.87] + \\ & 0.4[0 + 0.9 * 0]) \\ = \operatorname{argmax}_a (& 0, 0.741)\end{aligned}$$

Ostatecznie nową akcją wybraną dla stanu s_2 jest akcja a_1 , czyli:

$$\pi_1(s_2) = a_1$$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Wyniki działania algorytmu iteracji strategii po 1 kroku:

- Strategia:

- $\pi_1(s_0) = a_0.$

- $\pi_1(s_1) = a_0.$

- $\pi_1(s_2) = a_1.$

- Wartość funkcji:

- $v_{\pi_1}(s_0) = 2.83.$

- $v_{\pi_1}(s_1) = 6.49.$

- $v_{\pi_1}(s_2) = 3.47.$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - przykład liczbowy cd.

Wyniki działania algorytmu iteracji strategii po 3 kroku:

- Optymalna strategia:

- $\pi_3(s_0) = a_1.$

- $\pi_3(s_1) = a_0.$

- $\pi_3(s_2) = a_1.$

- Wartość funkcji:

- $v_{\pi_3}(s_0) = 3.79.$

- $v_{\pi_3}(s_1) = 7.3.$

- $v_{\pi_3}(s_2) = 4.21.$



Uczenie pasywne

Iteracyjne doskonalenie strategii (ang. *Policy Iteration*) - ćwiczenie

Implementacja algorytmu iteracyjnego doskonalenia strategii w pliku *PolicyIteration.py*:

- implementacja algorytmu oceny strategii z wykorzystaniem obliczeń w miejscu (funkcja *policy_eval_in_place*),
- implementacja algorytmu poprawy strategii (funkcja *policy_improvement*),
- implementacja algorytmu iteracyjnego doskonalenia strategii (funkcja *policy_iteration*).



Uczenie pasywne

Iteracja funkcji wartości



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Cel:

- Optymalizacja wyznaczania najlepszej strategii.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Cel:

- Optymalizacja wyznaczania najlepszej strategii.

Rozwiązanie:

- Połączenie algorytmów oceny strategii i poprawy strategii w pojedynczej aktualizacji funkcji wartości stanu.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Cel:

- Optymalizacja wyznaczania najlepszej strategii.

Rozwiązanie:

- Połączenie algorytmów oceny strategii i poprawy strategii w pojedynczej aktualizacji funkcji wartości stanu.

Optymalna strategia:

- Strategia określana jest tylko jeden raz, na końcu działania algorytmu, na podstawie otrzymanej funkcji wartości.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Iteracja wartości (ang. *Value Iteration*)

Wejście:

- θ - dokładność obliczania funkcji wartości.

Wyjście:

- $\pi(s)$ - optymalna strategia π .

Inicjalizacja tablicy wartości stanów $V(s)$ losowymi wartościami, za wyjątkiem stanu końcowego, któremu przypisana jest wartość 0.

Licz:

$$\Delta \leftarrow 0$$

Dla każdego $s \in S$:

$$v \leftarrow V(s)$$

$$V(s) \leftarrow \max_a \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma V(s')]$$

$$\Delta \leftarrow \max(\Delta, |v - V(s)|)$$

Dopóki $\Delta > \theta$

Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Iteracja wartości (ang. *Value Iteration*)

Wyznaczenie optymalnej strategii, $\pi \approx \pi_*$, na podstawie obliczonej funkcji wartości, z wykorzystaniem algorytmu zachłannego:

$$\pi(s) = \operatorname{argmax}_a \sum_{s'} \sum_r p(s', r | s, a) [r + \gamma V(s')] \quad (9)$$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*)

Value Iteration

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation

Initialize $V(s)$, for all $s \in S^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in S$:

$v \leftarrow V(s)$

$V(s) \leftarrow \max_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

Until $\Delta > \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that

$$\pi(s) = \operatorname{argmax}_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')] \quad (10)$$

Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy

Model środowiska:



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Wejście:

- Stany:

- $S = \{s_0, s_1, s_2\}$.

- Akcje:

- $A_{s_0} = \{a_0, a_1\}$.

- $A_{s_1} = \{a_0, a_1\}$.

- $A_{s_2} = \{a_0, a_1\}$.

- Funkcja wartości:

- $V(s_0) = 0$.

- $V(s_1) = 0$.

- $V(s_2) = 0$.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Wejście:

- Prawdopodobieństwo przejścia:

- $p(s_2|s_0, a_0) = 0.5$, $p(s_0|s_0, a_0) = 0.5$, $p(s_2|s_0, a_1) = 1$.
- $p(s_0|s_1, a_0) = 0.7$, $p(s_1|s_1, a_0) = 0.1$, $p(s_2|s_1, a_0) = 0.2$,
 $p(s_1|s_1, a_1) = 0.95$, $p(s_2|s_1, a_1) = 0.05$.
- $p(s_0|s_2, a_0) = 0.4$, $p(s_2|s_2, a_0) = 0.6$, $p(s_0|s_2, a_1) = 0.3$,
 $p(s_1|s_2, a_1) = 0.3$, $p(s_2|s_2, a_1) = 0.4$.

- Nagroda:

- $r(s_1, a_0, s_0) = 5$.
- $r(s_2, a_1, s_0) = -1$.
- W pozostałych przypadkach: $r = 0$.

- Discount factor:

- $\gamma = 0.9$.



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Aktualizowanie funkcji wartości w stanie s_0 :

$$V(s_0) = \max_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$$

Potrzebne dane:

- $p(s_2|s_0, a_0) = 0.5$, $p(s_0|s_0, a_0) = 0.5$, $p(s_2|s_0, a_1) = 1$.
- $r = 0$.
- $V(s_0) = 0.0$, $V(s_1) = 0.0$, $V(s_2) = 0.0$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Aktualizowanie funkcji wartości w stanie s_0 :

$$V(s_0) = \max_a (p(s_2|s_0, a_0)[r + \gamma V(s_2)] + \\ p(s_0|s_0, a_0)[r + \gamma V(s_0)], \\ p(s_2|s_0, a_1)[r + \gamma V(s_2)])$$

Po podstawieniu wartości otrzymujemy:

$$V(s_0) = \max_a (0.5[0 + 0.9 * 0] + \\ 0.5[0 + 0.9 * 0], \\ 1[0 + 0.9 * 0]) \\ = \max_a (0, 0)$$

Ostatecznie:

$$V(s_0) = 0$$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Aktualizowanie funkcji wartości w stanie s_1 :

$$V(s_1) = \max_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$$

Potrzebne dane:

- $p(s_0|s_1, a_0) = 0.7$, $p(s_1|s_1, a_0) = 0.1$, $p(s_2|s_1, a_0) = 0.2$,
 $p(s_1|s_1, a_1) = 0.95$, $p(s_2|s_1, a_1) = 0.05$.
- $r(s_1, a_0, s_0) = 5$, $r = 0$.
- $V(s_0) = 0.0$, $V(s_1) = 0.0$, $V(s_2) = 0.0$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Poprawa strategii w stanie s_1 :

$$\begin{aligned} V(s_1) = \max_a & (p(s_0|s_1, a_0)[r + \gamma V(s_0)] + \\ & p(s_1|s_1, a_0)[r + \gamma V(s_1)] + \\ & p(s_2|s_1, a_0)[r + \gamma V(s_2)], \\ & p(s_1|s_1, a_1)[r + \gamma V(s_1)] + \\ & p(s_2|s_1, a_1)[r + \gamma V(s_2)]) \end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned} V(s_1) = \max_a & (0.7[5 + 0.9 * 0] + \\ & 0.1[0 + 0.9 * 0] + \\ & 0.2[0 + 0.9 * 0], \\ & 0.95[0 + 0.9 * 0] + \\ & 0.05[0 + 0.9 * 0]) \\ = \max_a & (3.5, 0) \end{aligned}$$

Ostatecznie:

$$V(s_1) = 3.5$$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Aktualizowanie funkcji wartości w stanie s_2 :

$$V(s_2) = \max_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$$

Potrzebne dane:

- $p(s_0|s_2, a_0) = 0.4$, $p(s_2|s_2, a_0) = 0.6$, $p(s_0|s_2, a_1) = 0.3$,
 $p(s_1|s_2, a_1) = 0.3$, $p(s_2|s_2, a_1) = 0.4$.
- $r(s_2, a_1, s_0) = -1$, $r = 0$.
- $V(s_0) = 0.0$, $V(s_1) = 3.5$, $V(s_2) = 0.0$



Uczenie pasywne

Iteracja wartości - przykład liczbowy cd.

Poprawa strategii w stanie s_2 :

$$\begin{aligned}
 V(s_2) = \operatorname{argmax}_a (& p(s_0|s_2, a_0)[r + \gamma V(s_0)] + \\
 & p(s_2|s_2, a_0)[r + \gamma V(s_2)], \\
 & p(s_0|s_2, a_1)[r + \gamma V(s_0)] + \\
 & p(s_1|s_2, a_1)[r + \gamma V(s_1)] + \\
 & p(s_2|s_2, a_1)[r + \gamma V(s_2)])
 \end{aligned}$$

Po podstawieniu wartości otrzymujemy:

$$\begin{aligned}
 V(s_2) = \max_a (& 0.4[0 + 0.9 * 0] + \\
 & 0.6[0 + 0.9 * 0], \\
 & 0.3[-1 + 0.9 * 0] + \\
 & 0.3[0 + 0.9 * 3.5] + \\
 & 0.4[0 + 0.9 * 0]) \\
 = \max_a (& 0, 0.645)
 \end{aligned}$$

Ostatecznie:

$$V(s_2) = 0.645$$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Wartości funkcji dla poszczególnych stanów wyznaczone po n krokach algorytmu iteracji wartości.

Stan	Start	Krok 1	Krok 5	Krok 25	Krok 37
$V(s_0)$	0	0	1.96	3.75	3.79
$V(s_1)$	0	3.5	5.6	7.26	7.3
$V(s_2)$	0	0.65	2.52	4.17	4.21



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - przykład liczbowy cd.

Po zakończeniu obliczania funkcji wartości, następnym krokiem jest wyznaczenie optymalnej strategii dla każdego ze stanów za pomocą wzoru:

$$\pi(s) = \operatorname{argmax}_a \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma V(s')]$$

Wyznaczona optymalna strategia wygląda następująco:

- $\pi(s_0) = a_1.$
- $\pi(s_1) = a_0.$
- $\pi(s_2) = a_1.$



Uczenie pasywne

Iteracja funkcji wartości (ang. *Value Iteration*) - ćwiczenie

Implementacja algorytmu iteracyjnego obliczania funkcji wartości *Value Iteration.py*:

- implementacja algorytmu iteracyjnego obliczania funkcji wartości (funkcja *value_iteration*).



Uczenie pasywne

Materiały uzupełniające

- Książka *Reinforcement Learning: An Introduction*, Richard S. Sutton and Andrew G. Barto, wydanie drugie, 2018.
 - Rozdział 4 - *Dynamic Programming*.
- Książka *Grokking Deep Reinforcement Learning*, Miguel Morales, 2020.
 - Rozdział 3 - *Balancing immediate and long-term goals*.
- Video *RL Course by David Silver - Lecture 3: Planning by Dynamic Programming*.
- Video *Artificial Intelligence Course by Pieter Abbeel - Lecture 8: Markov Decision Processes (MDPs)*.



Aproksymacja funkcji wartości stanu

