# Zaawansowane Metody Inteligencji Obliczeniowej

# Wykład 3: Procesy Decyzyjne Markowa

Michał Kempka     Marek Wydmuch     Bartosz Wieloch

14 marca 2022

Fundusze Europejskie
Polska Cyfrowa

Rzeczpospolita Polska

Unia Europejska
Europejski Fundusz Rozwoju Regionalnego

"Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI Tech)",
projekt finansowany ze środków Programu Operacyjnego Polska Cyfrowa POPC.03.02.00-00-0001/20

# Plan wykładu

# Agent i środowisko

- **Agent** — podejmuje decyzje (wykonuje **akcje**) i się uczy maksymalizować nagrody
- **Środowisko** — reaguje na akcje zmieniając swój **stan** oraz zwracając **nagrody**

$$S_0, A_0,\ R_1, S_1, A_1,\ R_2, S_2, A_2,\ R_{3,} S_3, A_3,\ \ldots$$

- po wykonaniu akcji $A_t$ agent dostaje nagrodę $R_{t+1}$ wraz z nowym stanem $S_{t+1}$ (różne konwencje w literaturze!)

# Procesy Decyzyjne Markowa (MDP)

Procesy Decyzyjne Markowa — będziemy stosowali skrót **MDP** (*ang.* ***M**arkov **D**ecision **P**rocesses*)

## Co to jest Proces Decyzyjny Markowa?

$$\begin{aligned} \mathcal{S} &\text{ — zbiór stanów} \\ \mathcal{A} &\text{ — zbiór akcji (gdy zależny od stanu: } \mathcal{A}(s)\text{)} \\ \mathcal{R} &\text{ — zbiór nagród} \\ p(s', r|s, a) &\text{ — funkcja przejść (definiuje dynamikę MDP)} \end{aligned}$$

- $p(s', r|s, a) = P(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a)$
  Jest to zwykła, **deterministyczna** funkcja 4 argumentowa.
- W MDP funkcja $p$ całkowicie określa dynamikę środowiska: prawdopodobieństwa $S_t$ oraz $R_t$ zależą tylko i wyłącznie od poprzedzającego stanu $S_{t-1}$ i akcji $A_{t-1}$ (własność Markowa)

# MDP

W MDP na podstawie 4-argumentowej funkcji $p$ możemy wyliczyć wszystko co dotyczy środowiska, np.:

- funkcja przejść stanów

$$p(s'|s,a) = P(S_t = s'|S_{t-1} = s, A_{t-1} = a) = \sum_{r \in \mathcal{R}} p(s', r|s,a)$$

- spodziewana nagroda dla pary stan-akcja

$$r(s,a) = \mathbb{E}\left[R_t|S_{t-1} = s, A_{t-1} = a\right] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|s,a)$$

- spodziewana nagroda dla pary stan-akcja-stan

$$r(s,a,s') = \mathbb{E}\left[R_t|S_{t-1} = s, A_{t-1} = a, S_t = s'\right] = \sum_{r \in \mathcal{R}} r \frac{p(s', r|s,a)}{p(s'|s,a)}$$

# Cele i nagrody

Celem agenta jest **maksymalizacja oczekiwanej całkowitej** nagrody którą dostanie (maksymalizacja nie tylko natychmiastowej nagrody ale **skumulowanej w długim okresie**).

Przykłady:

- robot uczący się chodzić: w każdym kroku nagroda proporcjonalna do przebytego dystansu
- robot wychodzący z labiryntu: $-1$ za każdy krok zanim znajdzie wyjście
- robot zbierający śmieci: $+1$ za każdy znaleziony i podniesiony śmieć ($-1$ za uderzenie w przeszkody itd.)

Nagroda **nie powinna** sugerować jak osiągnąć pożądany cel (np. w grze w szachy agent nie powinien być nagradzany za zbicie figury przeciwnika itp.)

# Spodziewany zysk

Formalnie chcemy maksymalizować oczekiwany zysk $G_t$ od chwili $t$.
$G_t$ to funkcja sekwencji nagród $R_{t+1}, R_{t+2}, R_{t+3}, \ldots$

- Suma nagród:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T,$$

  gdzie $T$ to ostatni krok.
  Ma sens gdy w problemie naturalnie występuje „ostatni" krok (np. koniec gry).
  Interakcje agent–środowisko można podzielić w takim wypadku na niezależne epizody.
- Suma **zdyskontowanych** nagród:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1},$$

  gdzie $0 \leqslant \gamma \leqslant 1$ jest współczynnikiem dyskontowym.
  - $\gamma = 0$ — agent jest „krótkowzroczny"
  - $\gamma \to 1$ — agent staje się „dalekowzroczny": przyszłe nagrody liczą się coraz bardziej

## Wzór rekurencyjny

Dla zdyskontowanych nagród mamy:

$$
\begin{aligned}
G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots \\
&= R_{t+1} + \gamma \left( R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ldots \right) \\
&= R_{t+1} + \gamma G_{t+1}
\end{aligned}
$$

# Polityka

Polityka $\pi$ — funkcja mapująca stan na prawdopodobieństwo wybrania każdej możliwej akcji

$$\pi(a|s) = P(A_t = a | S_t = s)$$

$\pi$ jest zwykłą funkcją dwuargumentową (a „$|$” tylko przypomina, że definiuje ona rozkład prawdopodobieństwa).

Metody uczenia ze wzmocnieniem określają jak polityka agenta zmienia się wraz z jego doświadczeniem.

# Funkcja wartości stanu

$v_\pi(s)$ — funkcja **wartości stanu** (inaczej: użyteczności stanu) dla polityki $\pi$ określa **oczekiwany zysk** gdy:

1. rozpoczynamy w stanie $s$
2. dalej podążamy za polityką $\pi$

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[G_t | S_t = s\right] \\
&= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s\right]
\end{aligned}
$$

$\mathbb{E}_\pi[\cdot]$ — oczekiwana wartość zmiennej losowej gdy agent stosuje politykę $\pi$

# Funkcja wartości akcji

$q_\pi(s, a)$ — funkcja wartości akcji $a$ podjętej w stanie $s$ dla polityki $\pi$ określa oczekiwany zysk gdy:

1. rozpoczynamy w stanie $s$
2. wykonujemy akcję $a$
3. dalej stosujemy politykę $\pi$

$$\begin{aligned} q_\pi(s, a) &= \mathbb{E}_\pi \left[ G_t | S_t = s, A_t = a \right] \\ &= \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right] \end{aligned}$$

# Równanie Bellmana

Zależność pomiędzy wartością stanu $s$ a wartością możliwych następnych stanów:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi\left[G_t | S_t = s\right] \\ &= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} | S_t = s\right] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)\left[r + \gamma \mathbb{E}_\pi\left[G_{t+1} | S_{t+1} = s'\right]\right] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a)\left[r + \gamma v_\pi(s')\right] \end{aligned}$$

Inaczej:

- suma $\left[r + \gamma v_\pi(s')\right]$
- po wszystkich możliwych $a$, $s'$ i $r$
- ważona prawdopodobieństwem $\pi(a|s)p(s', r|s, a)$

# Równanie Bellmana

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

$$v_\pi(s) = \sum_{a,s',r} \pi(a|s) p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

Z równania Bellmana można wyliczyć $v_\pi$ — mamy układ równań (dla każdego $s \in \mathcal{S}$)

# Równanie Bellmana

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

$$v_\pi(s) = \sum_{a,s',r} \pi(a|s) p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

Z równania Bellmana można wyliczyć $v_\pi$ — mamy układ równań (dla każdego $s \in \mathcal{S}$)

**Pytanie**

Czy łatwo jest wyliczyć $v_\pi(s)$ dla każdego $s \in \mathcal{S}$ korzystając z równania Bellmana?

# Optymalna polityka — funkcja wartości i akcji

Dla MDP można zdefiniować optymalną politykę — jest to polityka $\pi$ maksymalizująca $v_\pi(s)$ dla wszystkich stanów $s$.

Optymalna funkcja wartości:

$$v_*(s) = \max_\pi v_\pi(s)$$

dla każdego $s \in \mathcal{S}$
Optymalna funkcja akcji:

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

dla każdego $s \in \mathcal{S}$ oraz $a \in \mathcal{A}$
Zależność optymalnej funkcji akcji od funkcji wartości:

$$q_*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a\right]$$

## Optymalna polityka — równanie Bellmana

Równanie Bellmana dla optymalnej funkcji wartości stanu — bez odwołania do żadnej specyficznej polityki $\pi$.

$$
\begin{aligned}
v_*(s) &= \max_{a \in \mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}\left[G_t | S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}\left[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a\right] \\
&= \max_a \sum_{s',r} p(s', r | s, a)\left[r + \gamma v_*(s')\right]
\end{aligned}
$$

## Optymalna polityka — równanie Bellmana

Równanie Bellmana dla optymalnej funkcji wartości stanu — bez odwołania do żadnej specyficznej polityki $\pi$.

$$\begin{aligned}
v_*(s) &= \max_{a\in\mathcal{A}(s)} q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}\left[G_t | S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}\left[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a\right] \\
&= \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_*(s')\right]
\end{aligned}$$

**Pytanie**

Czy łatwo jest wyliczyć $v_*(s)$ dla każdego $s \in \mathcal{S}$ korzystając z powyższego równania Bellmana?

# Optymalna polityka – równanie Bellmana

Równanie Bellmana dla optymalnej funkcji wartości akcji:

$$\begin{aligned} q_*(s,a) &= \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1},a') | S_t = s, A_t = a\right] \\ &= \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} q_*(s',a')\right] \end{aligned}$$

# Optymalna polityka — rozwiązanie

Mając daną funkcję wartości stanu $v_*(s)$ optymalna polityka to taka która przypisuje niezerowe prawdopodobieństwo tylko i wyłącznie dla akcji które mają maksymalną **oczekiwaną** wartość:

$$\pi_*(s|a) > 0 \Rightarrow a \in \arg\max_a \sum_{s',r} p(s', r|s, a)v_*(s')$$

Mając daną funkcję wartości akcji $q_*(s, a)$ optymalna polityka to:

$$\pi_*(s|a) > 0 \Rightarrow a \in \arg\max_a q_*(s, a)$$

## Optymalna polityka — rozwiązanie

Mając daną funkcję wartości stanu $v_*(s)$ optymalna polityka to taka która przypisuje niezerowe prawdopodobieństwo tylko i wyłącznie dla akcji które mają maksymalną **oczekiwaną** wartość:

$$\pi_*(s|a) > 0 \Rightarrow a \in \arg\max_a \sum_{s',r} p(s', r|s, a) v_*(s')$$

Mając daną funkcję wartości akcji $q_*(s, a)$ optymalna polityka to:

$$\pi_*(s|a) > 0 \Rightarrow a \in \arg\max_a q_*(s, a)$$

**Pytanie**

Wyznaczenie optymalnej polityki **wymaga modelu świata** dla metody opartej na funkcji wartości stanu, wartości akcji czy dla obu?

# Plan wykładu

1. Procesy Decyzyjne Markowa

2. Programowanie dynamiczne

# Ewaluacja polityki

Ewaluacja polityki (*ang. Policy Evaluation*) — proces wyliczania funkcji wartości stanu $v_\pi(s)$ dla konkretnej polityki $\pi$.

$$\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[G_t | S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} | S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s\right] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a)\left[r + \gamma v_\pi(s')\right]
\end{aligned}$$

- gdy dynamika problemu całkowicie znana (MDP) to mamy układ $|\mathcal{S}|$ równań z $|\mathcal{S}|$ niewiadomymi
- rozwiązanie jest proste (aczkolwiek może być pracochłonne!) — mamy gotowe metody rozwiązywania układów równań
- alternatywa: **metody iteracyjne**

# Ewaluacja polityki — metoda iteracyjna

W metodzie iteracyjnej rozważamy **sekwencję przybliżeń** funkcji wartości: $v_0, v_1, v_2, \ldots$

- początkowe przybliżenie wybrane arbitralnie (np. same zera)
- każde kolejne przybliżenie uzyskane z równania Bellmana:

$$\begin{aligned} v_{k+1}(s) &= \mathbb{E}_\pi \left[ R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s \right] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[ r + \gamma v_k(s') \right] \end{aligned}$$

  dla wszystkich $s \in \mathcal{S}$
- sekwencja $v_k$ zbiega do $v_\pi$ dla $k \to \infty$
- implementacja powyższej aktualizacji wymaga dwóch tablic: dla $v_{k+1}(s)$ oraz dla $v_k(s)$
- aktualizacja „w miejscu" — mamy tylko jedną tablicę, aktualizacja wartości dla stanu $s$ może zależeć zarówno od starych jak i już zaktualizowanych wartości (również zbiega do $v_\pi$)

# Ewaluacja polityki — algorytm

**Algorytm 1:** Pseudokod dla algorytmu **ewaluacji polityki** (ang. Policy Evaluation)

**Argumenty:**
Polityka $\pi(s)$ dla wszystkich $s \in \mathcal{S}$
$\Delta_{min} \leftarrow$ mała dodatnia liczba

**Inicjalizacja:**
Dowolnie zainicjalizowane $V(s)$ (np. $V(s) = 0$) – estymat wartości stanów $v(s)$

**repeat**
    $\Delta \leftarrow 0$
    **for** $s \in \mathcal{S}$ **do**
        $v \leftarrow V(s)$
        $V(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
**until** $\Delta < \Delta_{min}$;

# Ulepszenie polityki

Celem ewaluacji polityki jest znalezienie lepszej polityki :)

- wiemy jak dobrze jest postępować zgodnie z daną polityką $\pi$
- czy warto w stanie $s$ użyć innej akcji $a$ i dalej podążać już zgodnie z $\pi$?
- wartość takiej akcji jest równa:

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a\right] \\ &= \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right] \end{aligned}$$

- jeśli $q_\pi(s,a) > v_\pi(s)$
  to opłaca się zmienić politykę dla stanu $s$!
  (mamy gwarancję poprawy polityki)

# Algorytm iteracji polityki (*ang. Policy Iteration*)

**Algorytm 2:** Pseudokod dla algorytmu **iteracji polityki** (ang. Policy Iteration)

**1. Inicjalizacja:**
Dowolnie zainicjalizowane $V(s)$ (np. $V(s) = 0$) i $\pi(s)$ dla wszystkich $s \in \mathcal{S}$
$\Delta_{min} \leftarrow$ mała dodatnia liczba

**2. Ewaluacja polityki:**
**repeat**
  $\Delta \leftarrow 0$
  **for** $s \in \mathcal{S}$ **do**
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
**until** $\Delta < \Delta_{min}$;

**3. Polepszenie polityki:**
$Stop \leftarrow True$
**for** $s \in \mathcal{S}$ **do**
  $a \leftarrow \pi(s)$
  $\pi(s) \leftarrow$
    $\arg\max_a \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$
  **if** $a \neq \pi(s)$ **then** $Stop \leftarrow False$
**if** $Stop$ **then return** $\pi$ **else goto 2**

# Algorytm iteracji polityki — uwagi

- Jeśli $\theta$ będzie zbyt duże to procedura ewaluacji polityki (krok 2) zakończy się zbyt szybko
  - nawet bardzo niewielka poprawa dokładności $V(s)$ może spowodować wybranie przez $\arg\max$ innej akcji w procedurze ulepszenia polityki (krok 3)
  - może to skutkować ustabilizowaniem polityki $\pi$ na suboptymalnym rozwiązaniu
- W procedurze ulepszenia polityki, polityka staje się stabilna gdy **zbiór** najlepszych akcji (maksymalizujących wartość akcji) się nie zmienia.
  Jeśli $\arg\max$ zwraca tylko jedną arbitralną akcję może dojść do niekończącego przełączania się między dwoma równie dobrymi akcjami.
- W procedurze ewaluacji polityki zaczynamy od aktualnego oszacowania $V$ z poprzedniego kroku — szybsza zbieżność.

## Algorytm iteracji wartości (*ang. Value Iteration*)

Algorytm iteracji wartości to specyficzny przypadek algorytmu iteracji polityki:

- **jedna** iteracja ewaluacji polityki
- ulepszenie polityki

W praktyce nie wyznacza się jawnie polityki w każdej iteracji — powyższe dwa kroki łączy się w jeden, a politykę wyznacza się dopiero na końcu.

# Algorytm iteracji wartości

**Algorytm 3:** Pseudokod dla algorytmu **iteracji wartości** (ang. Value Iteration)

**Inicjalizacja:**
Dowolnie zainicjalizowane $V(s)$ dla wszystkich $s \in \mathcal{S}$
$\Delta_{min} \leftarrow$ mała dodatnia liczba
**repeat**
    $\Delta \leftarrow 0$
    **for** $s_i \in \mathcal{S}$ **do**
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_a \sum_{s',r} p((s', r|s, a)[r + \gamma V(s')]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
**until** $\Delta < \Delta_{min}$;
**for** $s_i \in \mathcal{S}$ **do**
    $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p((s', r|s, a))[r + \gamma V(s')]$
**return** $\pi$

# Zmodyfikowany algorytm iteracji polityki

Wadą poprzednio omawianego algorytmu iteracji polityki jest koszt obliczeniowy:

- w głównej pętli występuje procedura ewaluacji polityki, która...
- też jest metodą iteracyjną.

Pomysł: zastosować przybliżoną wartość polityki $V(s)$ — wykonujemy $k$ kroków algorytmu iteracji wartości

# Zmodyfikowany algorytm iteracji polityki

**Algorytm 4:** Pseudokod dla algorytmu **iteracji polityki z** $k$

**1. Inicjalizacja:**

Dowolnie zainicjalizowane $V(s)$ (np. $V(s) = 0$) i $\pi(s)$ dla wszystkich $s \in \mathcal{S}$

$\Delta_{min} \leftarrow$ mała dodatnia liczba; $k \leftarrow$ dodatnia liczba całkowita

**2. Ewaluacja polityki:**

**for** $i = 0; i < k; i \leftarrow i + 1$ **do**

  **for** $s \in \mathcal{S}$ **do**

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

**3. Polepszenie polityki:**

$Stop \leftarrow True$

**for** $s \in \mathcal{S}$ **do**

  $a \leftarrow \pi(s)$

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

  **if** $a \neq \pi(s)$ **then** $Stop \leftarrow False$

**if** $Stop$ **then return** $\pi$ **else goto 2**

## Pytanie

Co się stanie gdy $k = 1$?

# Asynchroniczne programowanie dynamiczne

- Wadą omówionych metod jest to, że wymagają wykonywania pewnych operacji po wszystkich stanach.
- Dla dużej liczby stanów nawet pojedyncza iteracja bardzo kosztowna.
- Idea: nie przechodzić systematycznie po wszystkich stanach w każdej iteracji.
  - Aktualizacja wartości stanów w dowolnej kolejności
  - Dla niektórych stanów aktualizacje częściej, dla innych – rzadziej.
- Dla zbieżności nadal konieczna jest aktualizacja wartości dla wszystkich stanów!
- Ale możemy częściej aktualizować wartości dla stanów które interesują agenta.

# Bibliografia

[1] Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall, third edition.

[2] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Fundusze Europejskie
Polska Cyfrowa

Rzeczpospolita Polska

Unia Europejska
Europejski Fundusz Rozwoju Regionalnego

"Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI Tech)",
projekt finansowany ze środków Programu Operacyjnego Polska Cyfrowa POPC.03.02.00-00-0001/20