Problemy Decyzyjne Markowa

na podstawie AIMA ch17 i slajdów S. Russel'a

Wojciech Jaśkowski

Instytut Informatyki,
Politechnika Poznańska

18 kwietnia 2013

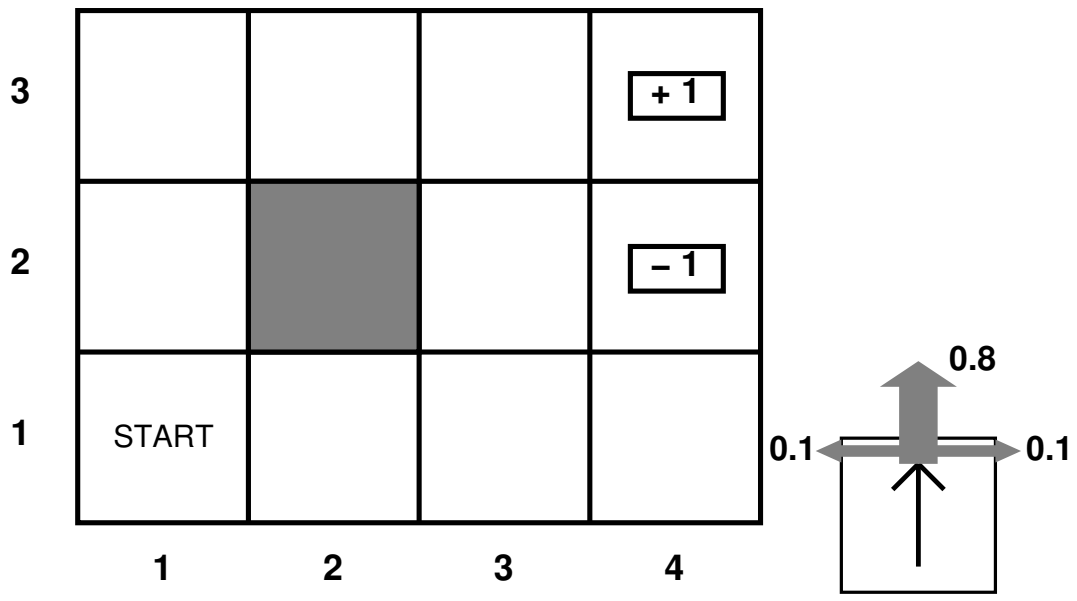
Sekwencyjne problemy decyzyjne

Cechy **sekwencyjnego problemu decyzyjnego**:

- ocena (użyteczność) agenta zależy od **sekwencji decyzji**, a nie od pojedynczej decyzji.
- **niepewność**, pomiary z sensorów

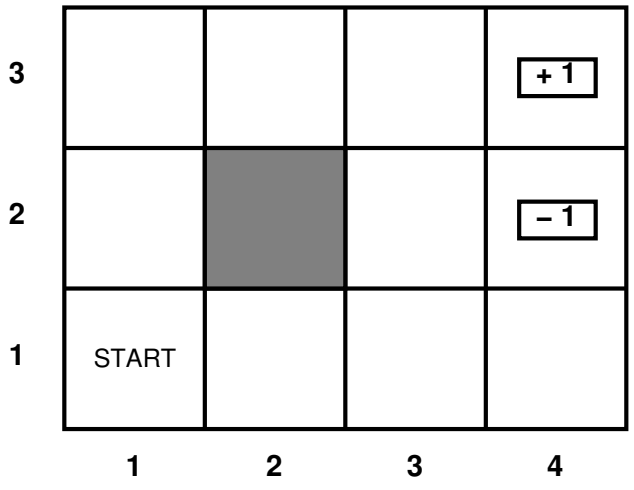
Problemy planowania i przeszukiwania — szczególny przypadek sekwencyjnych problemów decyzyjnych

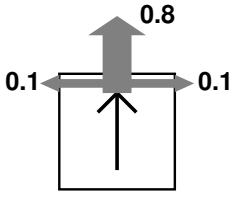
Środowisko



- Agent rozpoczyna na polu „Start”.
- W każdym kroku wykonuje akcję Góra, Dół, Lewo, Prawo.
- Interakcja ze środowiskiem kończy się, gdy agent dotrze do pól +1 lub -1.
- „Nagroda” za odwiedzenie pola wynosi +1,-1 lub -0.04
- Agent zawsze wie gdzie jest.

Cechy środowiska





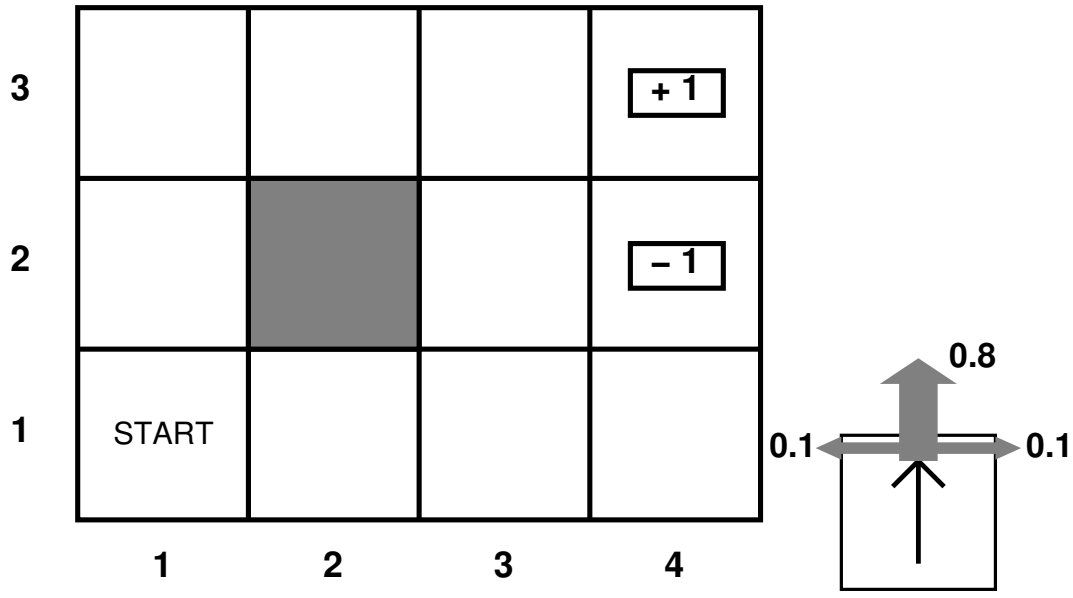
Środowisko jest: [zadanie 1]

1. Całkowicie vs. częściowo obserwowalne?
2. Deterministyczne vs. stochastyczne?
3. Epizodyczne vs. sekwencyjne?
4. Statyczne vs. dynamiczne vs. semidynamiczne?
5. Dyskretne vs. ciągłe?
6. Znane vs. nieznane?

Środowisko deterministyczne $\Longrightarrow$ proste rozwiązanie GGPPP.

- Ile wynosi prawd. dotarcia do $+1$? [zadanie 2]

Proces decyzyjny Markowa (MDP)



MDP — sekwencyjny proces decyzyjny dla środowiska:

1. całkowicie obserwowalnego
2. stochastycznego
3. z „markowskim” modelem przejść
4. z addytywną funkcja nagrody

Proces decyzyjny Markowa (MDP) — definicja

Elementy MDP:

- **Stany** $s \in S$, stan początkowy $s_0 \in S$ i stany terminalne
- **Akcje** $a \in A$
- **Model (przejść)** $T(s, a, s') \equiv P(s'|s, a)$ — prawd., że akcja a w stanie s prowadzi do stanu s'.
  - własność Markowa
- **Funkcja nagrody** (ang. reward) $R(s)$
  - $R(s) = \begin{cases} -0,04 & \text{(kara) dla stanów nieterminalnych} \\ \pm 1 & \text{dla stanów terminalnych} \end{cases}$
    - Można też uogólnić nagrodę do $R(s, a)$ lub $R(s, a, s')$, ale nie zmienia to podstawowych cech problemu.
- **Użyteczność** agenta jest określona przez funkcję nagrody (np. suma nagród w czasie życia).

Proces decyzyjny Markowa (MDP) — definicja

Elementy MDP:

- **Stany** $s \in S$, stan początkowy $s_0 \in S$ i stany terminalne
- **Akcje** $a \in A$
- **Model (przejść)** $T(s, a, s') \equiv P(s'|s, a)$ — prawd., że akcja a w stanie s prowadzi do stanu s'.
  - własność Markowa
- **Funkcja nagrody** (ang. reward) $R(s)$
  - $R(s) = \begin{cases} -0,04 & \text{(kara) dla stanów nieterminalnych} \\ \pm 1 & \text{dla stanów terminalnych} \end{cases}$
    - Można też uogólnić nagrodę do $R(s, a)$ lub $R(s, a, s')$, ale nie zmienia to podstawowych cech problemu.
- **Użyteczność** agenta jest określona przez funkcję nagrody (np. suma nagród w czasie życia).

Proces decyzyjny Markowa (MDP) — definicja

Elementy MDP:

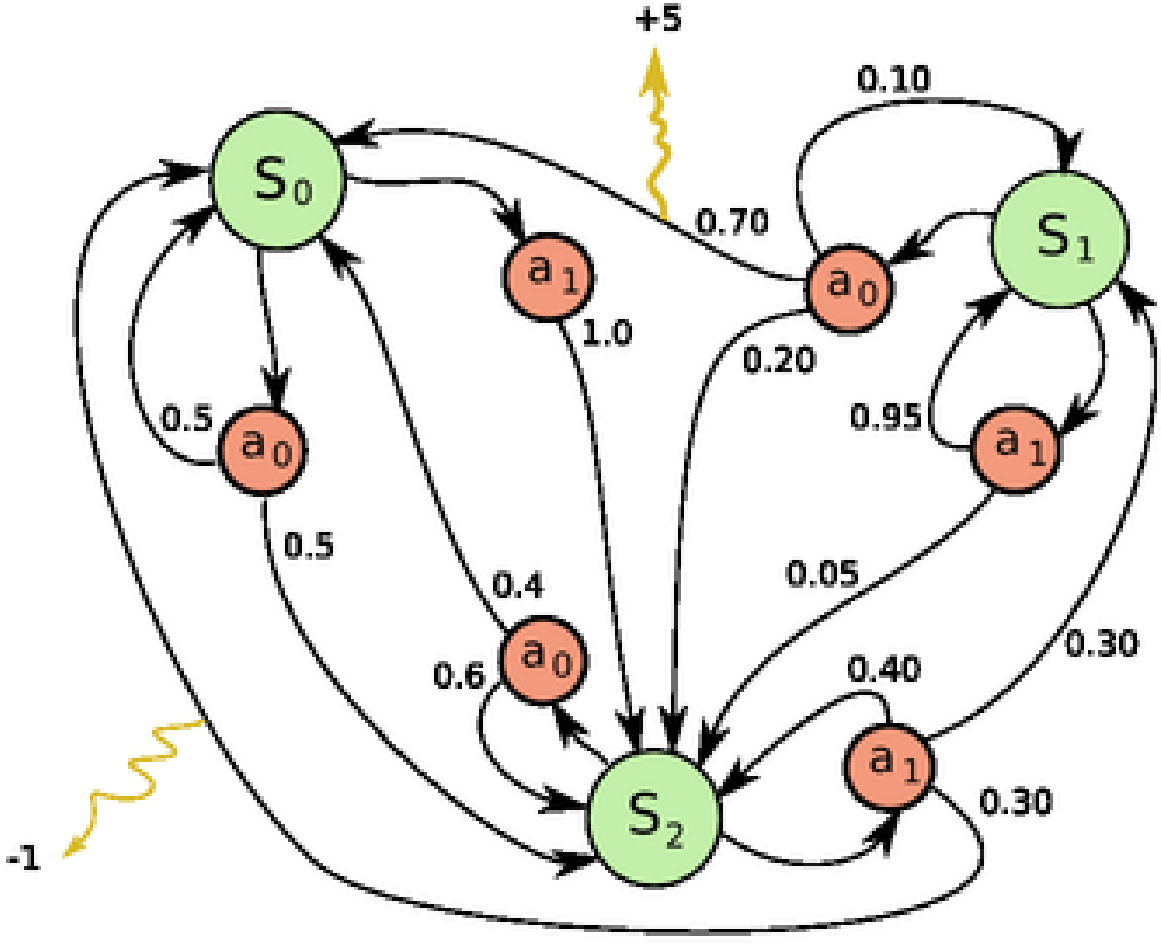
- **Stany** $s \in S$, stan początkowy $s_0 \in S$ i stany terminalne
- **Akcje** $a \in A$
- **Model (przejść)** $T(s, a, s') \equiv P(s'|s, a)$ — prawd., że akcja a w stanie s prowadzi do stanu s'.
  - własność Markowa
- **Funkcja nagrody** (ang. reward) $R(s)$
  - $R(s) = \begin{cases} -0,04 & \text{(kara) dla stanów nieterminalnych} \\ \pm 1 & \text{dla stanów terminalnych} \end{cases}$
    - Można też uogólnić nagrodę do $R(s, a)$ lub $R(s, a, s')$, ale nie zmienia to podstawowych cech problemu.
- **Użyteczność** agenta jest określona przez funkcję nagrody (np. suma nagród w czasie życia).

Proces decyzyjny Markowa (MDP) — definicja

Elementy MDP:

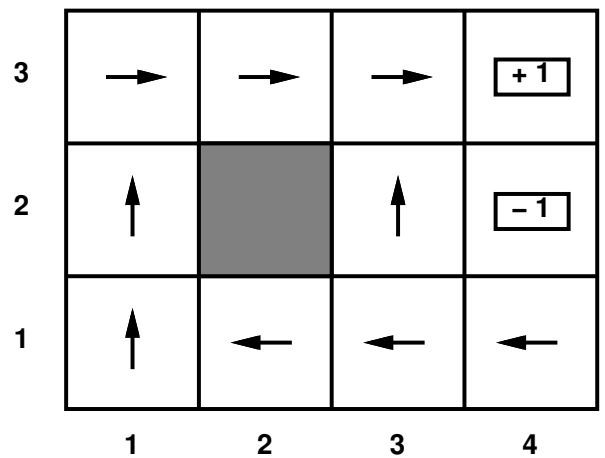
- **Stany** $s \in S$, stan początkowy $s_0 \in S$ i stany terminalne
- **Akcje** $a \in A$
- **Model (przejść)** $T(s, a, s') \equiv P(s'|s, a)$ — prawd., że akcja a w stanie s prowadzi do stanu s'.
  - własność Markowa
- **Funkcja nagrody** (ang. reward) $R(s)$
  - $R(s) = \begin{cases} -0,04 & \text{(kara) dla stanów nieterminalnych} \\ \pm 1 & \text{dla stanów terminalnych} \end{cases}$
    - Można też uogólnić nagrodę do $R(s, a)$ lub $R(s, a, s')$, ale nie zmienia to podstawowych cech problemu.
- **Użyteczność** agenta jest określona przez funkcję nagrody (np. suma nagród w czasie życia).

MDP — Przykład



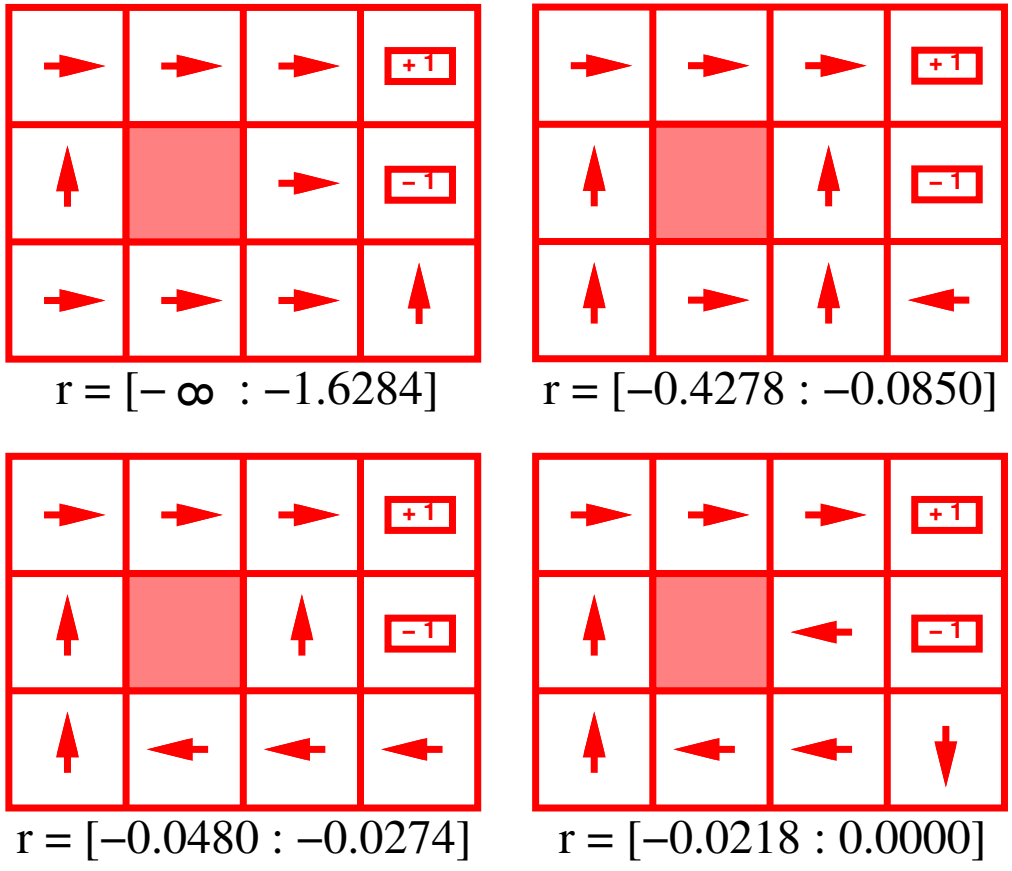
Rozwiązywanie MDP

- Cel w problemach przeszukiwania: znalezienie optymalnej sekwencji akcji.
- Cel w MDP: znalezienie **optymalnej polityki** (ang. optimal policy) $\pi(s)$
  - **polityka**: $\pi : S \rightarrow A$ (wybrana akcja dla każdego stanu s)
  - powód: nie wiadomo, które stany zostaną odwiedzone (środowisko jest niedeterministyczne)



- **optymalna polityka** π^* to polityka, która daje maksymalną **oczekiwaną** wartość funkcji użyteczności.

Ryzyko kary vs. nagroda



- r - kara dla stanów nieterminalnych

Ryzyko kary vs. nagroda c.d.

- Jak wygląda optymalna polityka dla $r > 0$ [zadanie 3]
- Kompromis pomiędzy **ryzykiem (kara)** a **nagrodą** jest cechą charakterystyczną MDP.
  - Nie występuje w problemach deterministycznych
  - Dlatego: MDP rozważane są w wielu dziedzinach, np.:
    - AI
    - badania operacyjne
    - ekonomia
    - teoria sterowania

Ryzyko kary vs. nagroda c.d.

- Jak wygląda optymalna polityka dla $r > 0$ [zadanie 3]
- Kompromis pomiędzy **ryzykiem (kara)** a **nagrodą** jest cechą charakterystyczną MDP.
  - Nie występuje w problemach deterministycznych
  - Dlatego: MDP rozważane są w wielu dziedzinach, np.:
    - AI
    - badania operacyjne
    - ekonomia
    - teoria sterowania

Użyteczność sekwencji stanów

- W MDP ocenie **podlega sekwencja stanów** $\rightarrow$ musimy zrozumieć preferencje pomiędzy sekwencjami stanów (nagród).
- Naturalne założenie: preferencje względem sekwencji stanów są **stacjonarne**, czyli:
  - $[s_1, s_2, \ldots] \succ [s'_1, s'_2, \ldots] \implies [s_0, s_1, s_2, \ldots] \succ [s_0, s'_1, s'_2, \ldots]$

Użyteczność sekwencji stanów

Twierdzenie

Przy założeniu stacjonarności sekwencji stanów istnieją tylko dwie możliwości, aby przyporządkować użyteczności do sekwencji (historii obserwacji) stanów:

- **Addytywna funkcja użyteczności**:
  - $U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + R(s_1) + R(s_2) + \ldots$
- **Zdyskontowana funkcja użyteczności** (ang. discounted utility function):
  - $U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$
  - gdzie $\gamma \in (0, 1)$ jest współczynnikiem dyskontowym (ang. discount factor)

Użyteczność sekwencji stanów

Twierdzenie

Przy założeniu stacjonarności sekwencji stanów istnieją tylko dwie możliwości, aby przyporządkować użyteczności do sekwencji (historii obserwacji) stanów:

- **Addytywna funkcja użyteczności**:
  - $U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + R(s_1) + R(s_2) + \ldots$
- **Zdyskontowana funkcja użyteczności** (ang. discounted utility function):
  - $U_h([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$
  - gdzie $\gamma \in (0, 1)$ jest współczynnikiem dyskontowym (ang. discount factor)

Użyteczności — problem nieskończoności

Problem: nieskończona przyszłość

$\Longrightarrow$ addytywna użyteczność może być nieskończona

$\Longrightarrow$ trudno porównywać dwie sekwencje stanów, gdy użyteczności obu wynoszą $+\infty$

Użyteczności — problem nieskończoności

Możliwe rozwiązania:

1. **Skończony horyzont:** koniec po liczbie kroków T
   - $\Longrightarrow$ polityka $\pi(s)$ jest niestacjonarna: tzn. zależy od pozostałego do końca czasu.
   - Przykład: pole [3,1] i T=3 vs. T=100.
2. Używanie **współczynnika dyskontowego** $\gamma < 1$
   - Jeśli $\forall_{s \in S} |R(s)| \leq R_{max}$, to $U([s_0, \ldots s_\infty]) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq R_{max}/(1-\gamma)$
   - Mniejsze $\gamma \Longrightarrow$ krótszy horyzont
3. **Stan(y) absorbujące** $\Longrightarrow$ z prawd. 1 agent w końcu zakończy działanie dla każdej polityki π
   - polityka, która zawsze prowadzi do stanu terminalnego, to **właściwa polityka**
   - możemy używać $\gamma = 1$
4. **Średnia nagroda** — maksym. średniej wypłaty na krok

Zwykle wybiera się opcję ze współczynnikiem dyskontowym.

Użyteczności — problem nieskończoności

Możliwe rozwiązania:

1. **Skończony horyzont:** koniec po liczbie kroków T
   - $\Longrightarrow$ polityka $\pi(s)$ jest niestacjonarna: tzn. zależy od pozostałego do końca czasu.
   - Przykład: pole [3,1] i T=3 vs. T=100.
2. Używanie **współczynnika dyskontowego** $\gamma < 1$
   - Jeśli $\forall_{s \in S} |R(s)| \leq R_{max}$, to $U([s_0, \ldots s_\infty]) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq R_{max}/(1-\gamma)$
   - Mniejsze $\gamma \Longrightarrow$ krótszy horyzont
3. **Stan(y) absorbujące** $\Longrightarrow$ z prawd. 1 agent w końcu zakończy działanie dla każdej polityki π
   - polityka, która zawsze prowadzi do stanu terminalnego, to **właściwa polityka**
   - możemy używać $\gamma = 1$
4. **Średnia nagroda** — maksym. średniej wypłaty na krok

Zwykle wybiera się opcję ze współczynnikiem dyskontowym.

Użyteczności — problem nieskończoności

Możliwe rozwiązania:

1. **Skończony horyzont:** koniec po liczbie kroków T
   - $\Longrightarrow$ polityka $\pi(s)$ jest niestacjonarna: tzn. zależy od pozostałego do końca czasu.
   - Przykład: pole [3,1] i T=3 vs. T=100.
2. Używanie **współczynnika dyskontowego** $\gamma < 1$
   - Jeśli $\forall_{s \in S} |R(s)| \leq R_{max}$, to $U([s_0, \ldots s_\infty]) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq R_{max}/(1-\gamma)$
   - Mniejsze $\gamma \Longrightarrow$ krótszy horyzont
3. **Stan(y) absorbujące** $\Longrightarrow$ z prawd. 1 agent w końcu zakończy działanie dla każdej polityki π
   - polityka, która zawsze prowadzi do stanu terminalnego, to **właściwa polityka**
   - możemy używać $\gamma = 1$
4. **Średnia nagroda** — maksym. średniej wypłaty na krok

Zwykle wybiera się opcję ze współczynnikiem dyskontowym.

Użyteczności — problem nieskończoności

Możliwe rozwiązania:

1. **Skończony horyzont:** koniec po liczbie kroków T
   - $\Longrightarrow$ polityka $\pi(s)$ jest niestacjonarna: tzn. zależy od pozostałego do końca czasu.
   - Przykład: pole [3,1] i T=3 vs. T=100.
2. Używanie **współczynnika dyskontowego** $\gamma < 1$
   - Jeśli $\forall_{s \in S} |R(s)| \leq R_{max}$, to $U([s_0, \ldots s_\infty]) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq R_{max}/(1-\gamma)$
   - Mniejsze $\gamma \Longrightarrow$ krótszy horyzont
3. **Stan(y) absorbujące** $\Longrightarrow$ z prawd. 1 agent w końcu zakończy działanie dla każdej polityki π
   - polityka, która zawsze prowadzi do stanu terminalnego, to **właściwa polityka**
   - możemy używać $\gamma = 1$
4. **Średnia nagroda** — maksym. średniej wypłaty na krok

Zwykle wybiera się opcję ze współczynnikiem dyskontowym.

Optymalna polityka i użyteczność stanów

Porównywanie polityk $\rightarrow$ porównywanie oczekiwanych wartości użyteczności sekwencji stanów.

Niech agent realizuje **politykę** π zaczynając od **stanu** s. Wtedy:

- S_t — zmienna losowa, oznaczająca stan osiągnięty w momencie t (czyli $S_0 = s$)
- Oczekiwana użyteczność polityki π rozpoczynając od stanu s:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

- Optymalna polityka π_s^* optymalizuje użyteczność (rozpoczynając od stanu s, czyli:

$$\pi_s^* = \mathrm{argmax}_{\pi} \mathrm{U}^{\pi}(\mathrm{s})$$

- Zadanie: mamy dwa stany $a \neq b$. Czy $\pi_a^* = \pi_b^*$? [zadanie 4]

Optymalna polityka i użyteczność stanów

Porównywanie polityk $\rightarrow$ porównywanie oczekiwanych wartości użyteczności sekwencji stanów.

Niech agent realizuje **politykę** π zaczynając od **stanu** s. Wtedy:

- S_t — zmienna losowa, oznaczająca stan osiągnięty w momencie t (czyli $S_0 = s$)
- Oczekiwana użyteczność polityki π rozpoczynając od stanu s:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

- Optymalna polityka π_s^* optymalizuje użyteczność (rozpoczynając od stanu s, czyli:

$$\pi_s^* = \mathrm{argmax}_{\pi} \mathrm{U}^{\pi}(\mathrm{s})$$

- Zadanie: mamy dwa stany $a \neq b$. Czy $\pi_a^* = \pi_b^*$? [zadanie 4]

Optymalna polityka i użyteczność stanów

Porównywanie polityk $\rightarrow$ porównywanie oczekiwanych wartości użyteczności sekwencji stanów.

Niech agent realizuje **politykę** π zaczynając od **stanu** s. Wtedy:

- S_t — zmienna losowa, oznaczająca stan osiągnięty w momencie t (czyli $S_0 = s$)
- Oczekiwana użyteczność polityki π rozpoczynając od stanu s:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

- Optymalna polityka π_s^* optymalizuje użyteczność (rozpoczynając od stanu s, czyli:

$$\pi_s^* = \mathrm{argmax}_{\pi} \mathrm{U}^{\pi}(\mathrm{s})$$

- Zadanie: mamy dwa stany $a \neq b$. Czy $\pi_a^* = \pi_b^*$? [zadanie 4]

Optymalna polityka i użyteczność stanów

Porównywanie polityk $\rightarrow$ porównywanie oczekiwanych wartości użyteczności sekwencji stanów.

Niech agent realizuje **politykę** π zaczynając od **stanu** s. Wtedy:

- S_t — zmienna losowa, oznaczająca stan osiągnięty w momencie t (czyli $S_0 = s$)
- Oczekiwana użyteczność polityki π rozpoczynając od stanu s:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

- Optymalna polityka π_s^* optymalizuje użyteczność (rozpoczynając od stanu s, czyli:

$$\pi_s^* = \mathrm{argmax}_{\pi} \mathrm{U}^{\pi}(\mathrm{s})$$

- Zadanie: mamy dwa stany $a \neq b$. Czy $\pi_a^* = \pi_b^*$?[zadanie 4]

Użyteczność stanów c.d.

- $U^{\pi^*}(s)$ jest użytecznością optymalnej polityki zaczynając od stanu s.
  - Będziemy ją oznaczać $U(s)$.
  - Czyli możemy ją interpretować jako **użyteczność stanu** s.
  - Porównanie R i U:
    - $R(s)$ — nagroda krótkoterminowa
    - $U(s)$ — nagroda długoterminowa

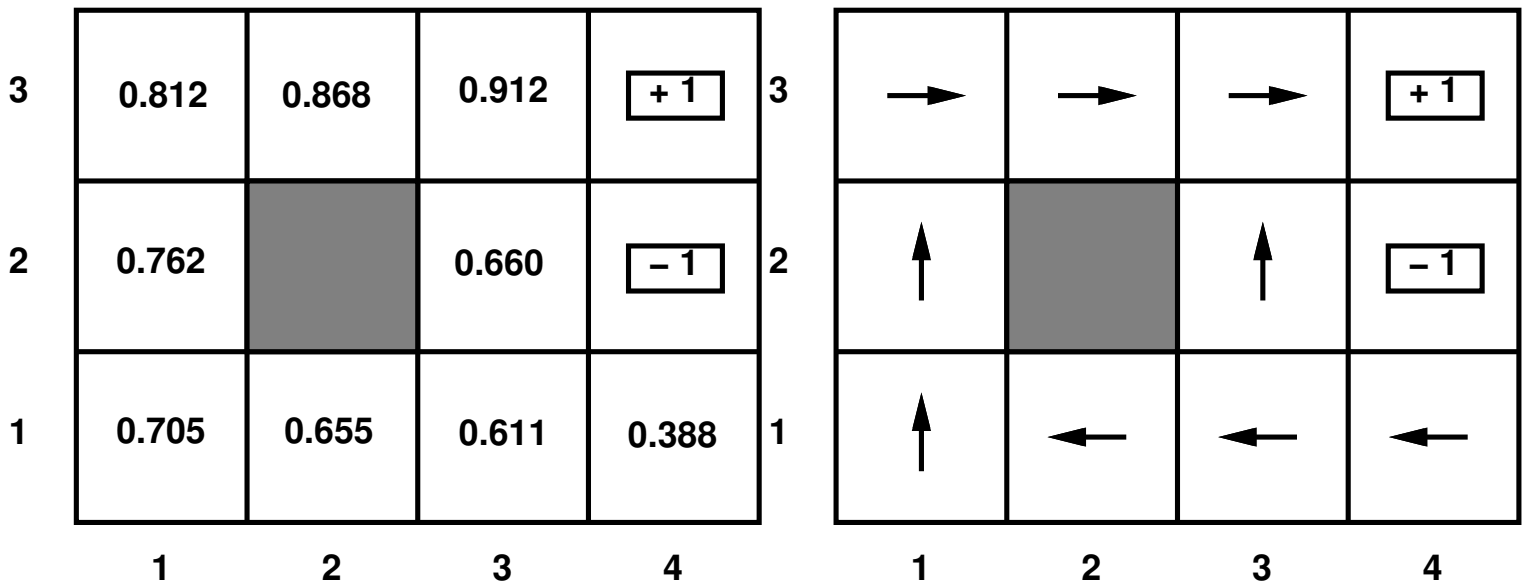
Użyteczność stanu — interpretacja

$U(s)$ jest oczekiwaną (zdyskontowaną) sumą nagród uzyskanych przez akcje zaczynające się w stanie s przy założeniu realizowania optymalnej polityki.

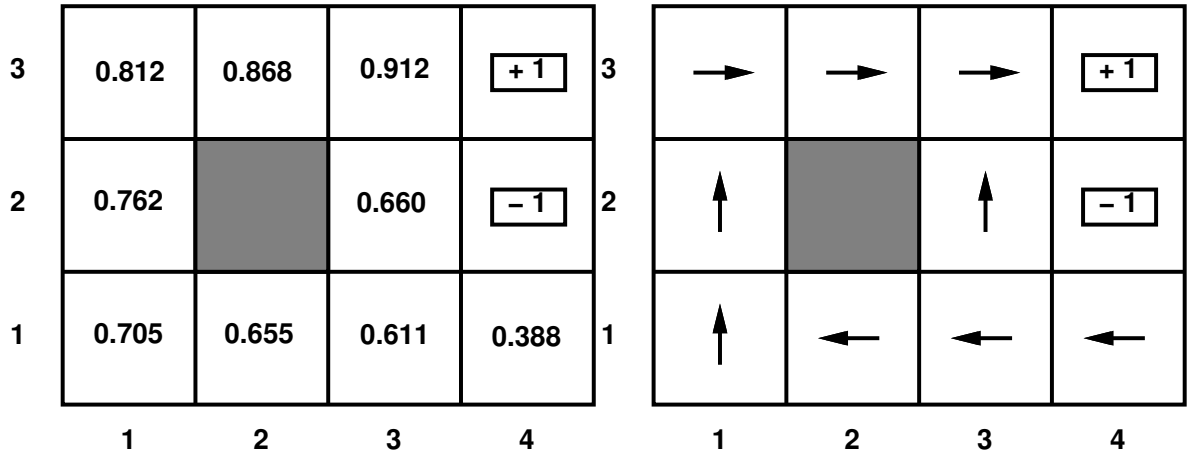
Użyteczność stanów (c.d)

- Użyteczności stanów jednoznacznie definiują politykę:
  - wystarczy znaleźć akcję, która ma maksymalną oczekiwaną użyteczność

$$\pi^*(s) = \text{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)U(s')$$



Optymalna akcja



$$\pi^*(s) = \text{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a) U(s')$$

- Dlaczego z $s_{3,1}$ idziemy w lewo a nie w górę?
- z $s_{3,1}$ w górę $= \langle 0.8, 0.1, 0.1 \rangle \times \langle 0.660, 0.655, 0.388 \rangle = 0.6323$
- z $s_{3,1}$ w lewo $= \langle 0.8, 0.1, 0.1 \rangle \times \langle 0.655, 0.611, 0.660 \rangle = 0.6511$

Programowanie dynamiczne: równanie Bellmana

Równanie Bellman'a (1957)

oczekiwana suma wypłat = aktualna wypłata
+ $\gamma \times$ oczekiwana suma wypłat po wybraniu najlepszej akcji:

$$U(s) = R(s) + \gamma \max_{a \in A} \sum_{s'} U(s') P(s'|s, a)$$

Przykład:

$$\begin{aligned} U(1,1) = -0.04 + \gamma \max\{ & \\ & 0.8 \times U(1,2) + 0.1 \times U(2,1) + 0.1 \times U(1,1), \\ & 0.9 \times U(1,1) + 0.1 \times U(1,2) \\ & 0.9 \times U(1,1) + 0.1 \times U(2,1), \\ & 0.8 \times U(2,1) + 0.1 \times U(1,2) + 0.1 \times U(1,1)\} \end{aligned}$$

Programowanie dynamiczne: równanie Bellmana

Równanie Bellman'a (1957)

oczekiwana suma wypłat = aktualna wypłata
+ $\gamma \times$ oczekiwana suma wypłat po wybraniu najlepszej akcji:

$$U(s) = R(s) + \gamma \max_{a \in A} \sum_{s'} U(s') P(s'|s, a)$$

Przykład:

$$\begin{aligned} U(1,1) = -0.04 + \gamma \max\{ & \\ & 0.8 \times U(1,2) + 0.1 \times U(2,1) + 0.1 \times U(1,1), \\ & 0.9 \times U(1,1) + 0.1 \times U(1,2) \\ & 0.9 \times U(1,1) + 0.1 \times U(2,1), \\ & 0.8 \times U(2,1) + 0.1 \times U(1,2) + 0.1 \times U(1,1)\} \end{aligned}$$

Układ równań Bellmana

$$\begin{cases} U(s_0) & = R(s_0) + \gamma \max_{a \in A} \sum_{s'} U(s') P(s'|s_0, a) \\ U(s_1) & = R(s_1) + \gamma \max_{a \in A} \sum_{s'} U(s') P(s'|s_1, a) \\ \vdots & \vdots \end{cases}$$

- 1 równanie na stan + n stanów $\implies$ n równań z n niewiadomymi.
- Równania są liniowe czy nieliniowe? [zadanie 5]

Algorytm iteracji wartości (Value Iteration)

Algorytm iteracji wartości:

1. Rozpocznij z dowolnymi wartościami użyteczności (np. losowymi)
2. Uaktualnij użyteczności zgodne z układem równań Bellmana (dla wszystkich s):

$$U_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} U_i(s') P(s'|s,a)$$

   - Uwaga: uaktualnienia wykonujemy synchronicznie (kopia tablicy U).
3. Jeśli osiągnęliśmy równowagę (brak zmian), to mamy **globalne** optimum

Algorytm iteracji wartości (Value Iteration)

Algorytm iteracji wartości:

1. Rozpocznij z dowolnymi wartościami użyteczności (np. losowymi)
2. Uaktualnij użyteczności zgodne z układem równań Bellmana (dla wszystkich s):

$$U_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} U_i(s') P(s'|s,a)$$

   - Uwaga: uaktualnienia wykonujemy synchronicznie (kopia tablicy U).

3. Jeśli osiągnęliśmy równowagę (brak zmian), to mamy **globalne** optimum

Algorytm iteracji wartości (Value Iteration)

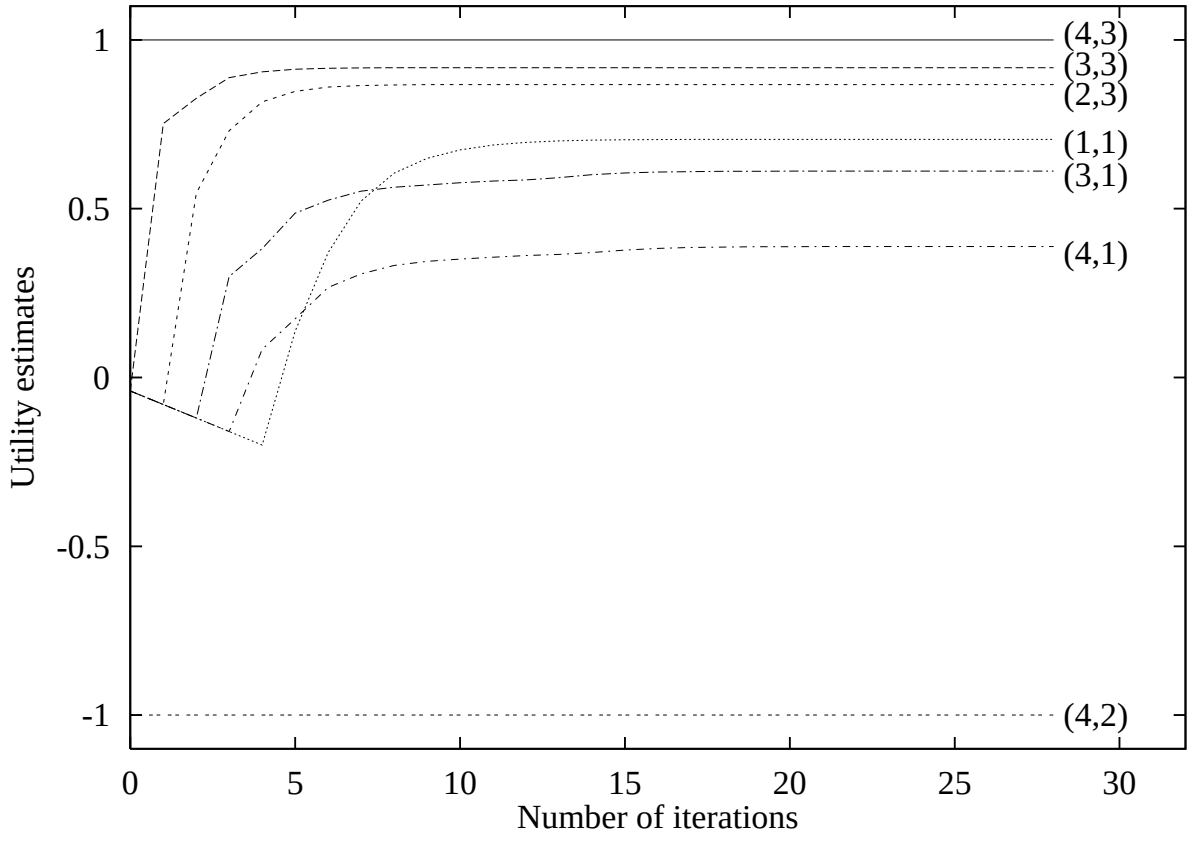
Algorytm iteracji wartości:

1. Rozpocznij z dowolnymi wartościami użyteczności (np. losowymi)
2. Uaktualnij użyteczności zgodne z układem równań Bellmana (dla wszystkich s):

$$U_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} U_i(s') P(s'|s,a)$$

   - Uwaga: uaktualnienia wykonujemy synchronicznie (kopia tablicy U).
3. Jeśli osiągnęliśmy równowagę (brak zmian), to mamy **globalne** optimum

Algorytm iteracji wartości — wykresy



Algorytm iteracji wartości — zbieżność

- Algorytm jest zbieżny do (globalnego) optimum
- Liczba wymaganych iteracji wynosi:

$$N = \lceil \log(2R_{\max}/\epsilon(1-\gamma))/\log(1/\gamma) \rceil ,$$

gdzie:

  - ϵ jest maksymalnym błędem pomiędzy obliczoną użytecznością stanu a użytecznością rzeczywistą
  - $R_{\max}$ jest maksymalną wartością nagrody

- Stosowane kryterium stopu:

$$\|U_{i+1} - U_i\| < \epsilon(1-\gamma)/\gamma$$

- **Optymalna polityka jest dostępna zanim wartości użyteczności zbiegną do idealnych**.
- Powyższe działa, gdy $\gamma < 1$. Jeśli $\gamma = 1$, trzeba innego kryterium stopu.

Algorytm iteracji polityki (ang. Policy Iteration)

- **Obserwacja**: można otrzymać optymalną politykę nawet gdy wartości funkcji użyteczności są niedokładne.
- **Pomysł**: Szukaj optymalnej polityki oraz wartości funkcji użyteczności równocześnie.

Algorytm (Howard, 1960):

1. $\pi \leftarrow$ dowolna polityka
2. powtarzaj dopóki π się zmienia:
   1. (**Ocena polityki**) Oblicz użyteczność $U^{\pi}(s)$ dla wszystkich stanów $s \in S$.
   2. (**Ulepszenie polityki**) Oblicz nową politykę π zakładając, że $U^{\pi}(s) = U(s)$ (użyteczności π są poprawne):

$$\pi(s) = \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)U(s')$$

Algorytm iteracji polityki (ang. Policy Iteration)

- **Obserwacja**: można otrzymać optymalną politykę nawet gdy wartości funkcji użyteczności są niedokładne.
- **Pomysł**: Szukaj optymalnej polityki oraz wartości funkcji użyteczności równocześnie.

Algorytm (Howard, 1960):

1. $\pi \leftarrow$ dowolna polityka
2. powtarzaj dopóki π się zmienia:
   1. (**Ocena polityki**) Oblicz użyteczność $U^{\pi}(s)$ dla wszystkich stanów $s \in S$.
   2. (**Ulepszenie polityki**) Oblicz nową politykę π zakładając, że $U^{\pi}(s) = U(s)$ (użyteczności π są poprawne):

$$\pi(s) = \text{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)U(s')$$

Algorytm iteracji polityki (ang. Policy Iteration)

- **Obserwacja**: można otrzymać optymalną politykę nawet gdy wartości funkcji użyteczności są niedokładne.
- **Pomysł**: Szukaj optymalnej polityki oraz wartości funkcji użyteczności równocześnie.

Algorytm (Howard, 1960):

1. $\pi \leftarrow$ dowolna polityka
2. powtarzaj dopóki π się zmienia:
   1. (**Ocena polityki**) Oblicz użyteczność $U^{\pi}(s)$ dla wszystkich stanów $s \in S$.
   2. (**Ulepszenie polityki**) Oblicz nową politykę π zakładając, że $U^{\pi}(s) = U(s)$ (użyteczności π są poprawne):

$$\pi(s) = \text{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)U(s')$$

Algorytm iteracji polityki (ang. Policy Iteration)

- **Obserwacja**: można otrzymać optymalną politykę nawet gdy wartości funkcji użyteczności są niedokładne.
- **Pomysł**: Szukaj optymalnej polityki oraz wartości funkcji użyteczności równocześnie.

Algorytm (Howard, 1960):

1. $\pi \leftarrow$ dowolna polityka
2. powtarzaj dopóki π się zmienia:
   1. (**Ocena polityki**) Oblicz użyteczność $U^{\pi}(s)$ dla wszystkich stanów $s \in S$.
   2. (**Ulepszenie polityki**) Oblicz nową politykę π zakładając, że $U^{\pi}(s) = U(s)$ (użyteczności π są poprawne):

$$\pi(s) = \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)U(s')$$

Algorytm iteracji polityki (ang. Policy Iteration)

- **Obserwacja**: można otrzymać optymalną politykę nawet gdy wartości funkcji użyteczności są niedokładne.
- **Pomysł**: Szukaj optymalnej polityki oraz wartości funkcji użyteczności równocześnie.

Algorytm (Howard, 1960):

1. $\pi \leftarrow$ dowolna polityka
2. powtarzaj dopóki π się zmienia:
   1. (**Ocena polityki**) Oblicz użyteczność $U^{\pi}(s)$ dla wszystkich stanów $s \in S$.
   2. (**Ulepszenie polityki**) Oblicz nową politykę π zakładając, że $U^{\pi}(s) = U(s)$ (użyteczności π są poprawne):

$$\pi(s) = \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a) U(s')$$

Algorytm iteracji polityki — ocena polityki

Aby obliczyć użyteczności polityki π wystarczy dla wszystkich stanów policzyć:

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} U^{\pi}(s') P(s'|s, \pi(s))$$

Tzn, mamy układ n liniowych równań i n niewiadomych. Można go rozwiązać w czasie: [zadanie 6]

- $O(n)$?
- $O(n^2)$?
- $O(n^3)$?

Algorytm iteracji polityki — ocena polityki

Aby obliczyć użyteczności polityki π wystarczy dla wszystkich stanów policzyć:

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} U^{\pi}(s') P(s'|s, \pi(s))$$

Tzn, mamy układ n liniowych równań i n niewiadomych. Można go rozwiązać w czasie: [zadanie 6]

- $O(n)$?
- $O(n^2)$?
- $O(n^3)$?

Zmodyfikowany algorytm iteracji polityki

Algorytm iteracji polityki:

- bardzo szybko zbiega do optimum, ale
- każdy jego krok jest kosztowny obliczeniowo, gdy stanów jest dużo:
  - $O(n^3)$ zaczyna być bolesne

Pomysł: Obliczmy więc iteracyjnie przybliżoną wartość $U(s)$

- (Przybliżoną) użyteczność $U(s)$ obliczamy wykonując k kroków algorytmu iteracji wartości (ze stałą polityką π) rozpoczynając od ostatnio znanego $U(s)$, czyli:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s')$$

- Zwykle szybciej zbiega znacznie szybciej niż „czysty” algorytm iteracji wartości lub iteracji polityki.

Zmodyfikowany algorytm iteracji polityki

Algorytm iteracji polityki:

- bardzo szybko zbiega do optimum, ale
- każdy jego krok jest kosztowny obliczeniowo, gdy stanów jest dużo:
  - $O(n^3)$ zaczyna być bolesne

Pomysł: Obliczmy więc iteracyjnie przybliżoną wartość $U(s)$

- (Przybliżoną) użyteczność $U(s)$ obliczamy wykonując k kroków algorytmu iteracji wartości (ze stałą polityką π) rozpoczynając od ostatnio znanego $U(s)$, czyli:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s')$$

- Zwykle szybciej zbiega znacznie szybciej niż „czysty” algorytm iteracji wartości lub iteracji polityki.

Zmodyfikowany algorytm iteracji polityki

Algorytm iteracji polityki:

- bardzo szybko zbiega do optimum, ale
- każdy jego krok jest kosztowny obliczeniowo, gdy stanów jest dużo:
  - $O(n^3)$ zaczyna być bolesne

Pomysł: Obliczmy więc iteracyjnie przybliżoną wartość $U(s)$

- (Przybliżoną) użyteczność $U(s)$ obliczamy wykonując k kroków algorytmu iteracji wartości (ze stałą polityką π) rozpoczynając od ostatnio znanego $U(s)$, czyli:

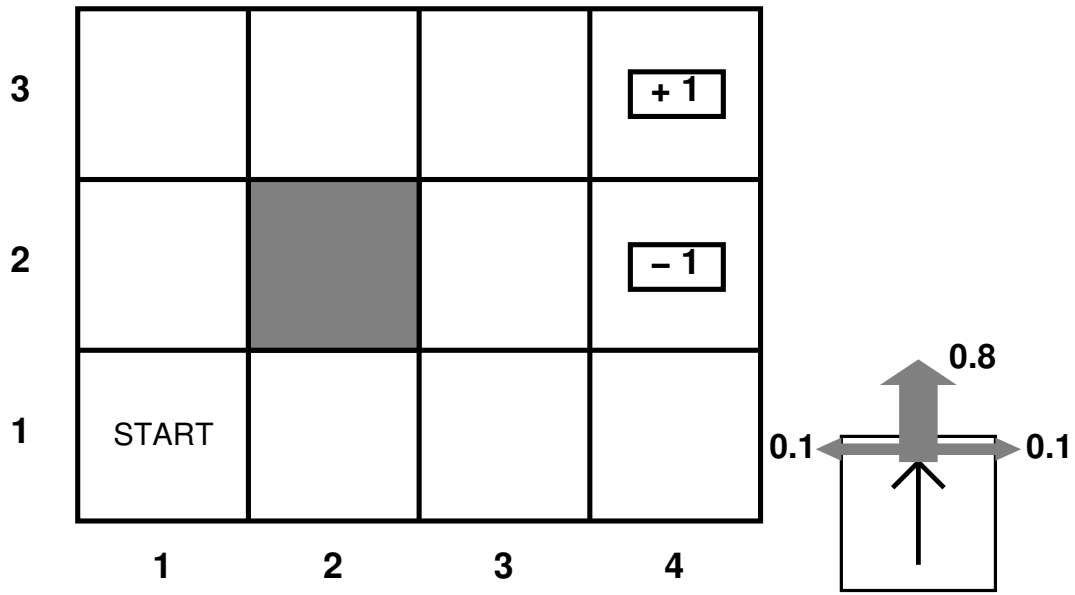
$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s')$$

- Zwykle szybciej zbiega znacznie szybciej niż „czysty” algorytm iteracji wartości lub iteracji polityki.

Rozszerzenia

- Do tej pory algorytmy równocześnie (synchronicznie) aktualizowały użyteczności poszczególnych stanów.
- Nie jest to konieczne. Można:
  - wybrać jakikolwiek podzbiór stanów i
  - zaaplikować do niego którąkolwiek aktualizację (ulepszenie polityki lub iteracje wartości)
  - $\Longrightarrow$ algorytm **Asynchronicznej Iteracji Polityki**
    - Pod pewnymi warunkami dot. początkowych użyteczności i początkowej polityki jest zbieżny
    - Umożliwia dobranie heurystyki wyboru stanów do aktualizacji, np. algorytm, który koncentruje się na ocenie użyteczności stanów, które z dużym prawd. mają szansę być odwiedzone.

Co zrobić, gdy agent nie wie gdzie jest?



Rzecz o częściowej obserwowalności

- Agent nie zna aktualnego stanu środowiska („nie wie gdzie jest") $\implies$ nie ma sensu mówić o polityce $\pi(s)$!
- Musi zbierać informacje i wnioskować na temat możliwych stanów środowiska (**stan przekonań**, ang. belief state = czyli rozkład prawd. nad stanami) $\implies$ **filtrowanie**.

POMDP

- POMDP:
  - zbiór stanów S (w tym stanów terminalnych)
  - zbiór akcji A
  - model przejść $P(s'|s,a)$
  - funkcja nagrody $R(s)$
  - model sensoryczny $P(e|s)$ czyli prawd., że agent notuje obserwację e będąc w stanie s.
    - W ogólności może przybrać formę $P(e|s,a,s')$
  - początkowy stan przekonań b_0 (nieznany jest jest stan początkowy s_0)

Stany przekonań

- Aktualizacja stanu przekonań — **problem filtrowania** (estymacji stanu systemu). Aktualizacja stanów przekonań:

$$b'(s') = \alpha P(e|s') \sum_s P(s'|s,a) b(s),$$

gdzie:

- $b(s)$ oznacza prawd., że agent jest w stanie s wg stanu przekonań b,
- α jest współczynnikiem normalizującym ($\sum_{s \in S} b(s) = 1$),
- e jest poczynioną obserwacją.

Stany przekonań

Twierdzenie (Astrom, 1965)

Optymalna polityka w POMDP jest funkcją $\pi : B \rightarrow A$, gdzie B jest zbiorem stanów przekonań). **Optymalna polityka nie zależy od aktualnego stanu, w którym jest agent.**

Jak zachowuje się agent realizujący politykę π?

Powtarza:

1. Wykonaj akcję $a = \pi(b)$, gdzie b jest aktualnym stanem przekonań agenta
2. Otrzymaj obserwację środowiska e
3. Zaktualizuj swój stan przekonań obliczając b' (filtrowanie)

Stany przekonań

Twierdzenie (Astrom, 1965)

Optymalna polityka w POMDP jest funkcją $\pi : B \rightarrow A$, gdzie B jest zbiorem stanów przekonań). **Optymalna polityka nie zależy od aktualnego stanu, w którym jest agent.**

Jak zachowuje się agent realizujący politykę π?

Powtarza:

1. Wykonaj akcję $a = \pi(b)$, gdzie b jest aktualnym stanem przekonań agenta
2. Otrzymaj obserwację środowiska e
3. Zaktualizuj swój stan przekonań obliczając b' (filtrowanie)

POMDP $\rightarrow$ MDP

Wniosek

Można przekształcić POMDP w MDP operujący w przestrzeni stanów przekonań, gdzie

- $P(b'|a,b)$ jest prawd., że nowym stanem przekonań będzie b' pod warunkiem, że aktualny stan przekonań to b i agent wykonuje akcję a.

$P(b'|a,b)$ można łatwo wyprowadzić.

POMDP $\rightarrow$ MDP

Ale, zauważmy, że przestrzeń stanów przekonań B:

- jest przestrzenią ciągłą (rozkłady prawd.),
- ma bardzo wiele wymiarów.
  - Jeśli mamy n stanów $\implies$ b jest n-wymiarowym wektorem liczb rzeczywistych

Istnieje **algorytm iteracji wartości** dla POMDP (1970), ale jest zbyt wolny nawet dla 4×3

- rozwiązywanie POMDP jest bardzo trudne obliczeniowo (PSPACE-trudne).

Istnieją algorytmy przybliżone oparte na **dynamicznych sieciach baysowskich**.

POMDP $\rightarrow$ MDP

Ale, zauważmy, że przestrzeń stanów przekonań B:

- jest przestrzenią ciągłą (rozkłady prawd.),
- ma bardzo wiele wymiarów.
  - Jeśli mamy n stanów $\implies$ b jest n-wymiarowym wektorem liczb rzeczywistych

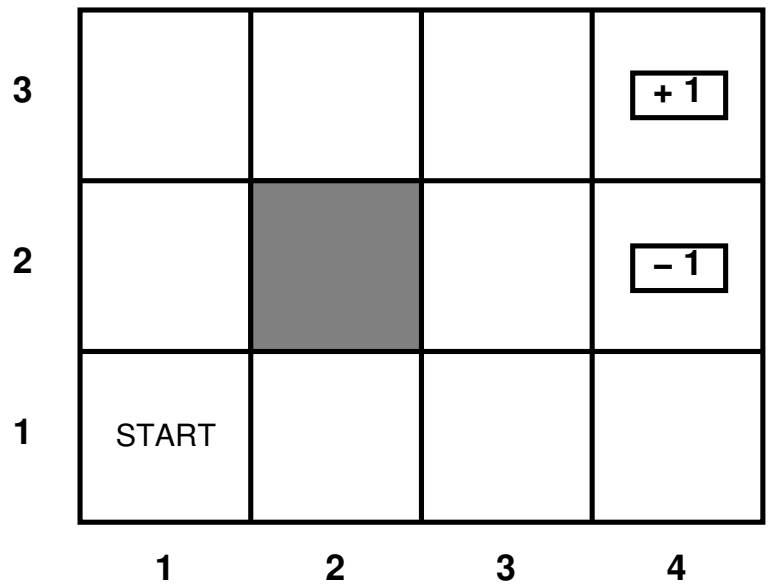
Istnieje **algorytm iteracji wartości** dla POMDP (1970), ale jest zbyt wolny nawet dla 4×3

- rozwiązywanie POMDP jest bardzo trudne obliczeniowo (PSPACE-trudne).

Istnieją algorytmy przybliżone oparte na **dynamicznych sieciach baysowskich**.

Zadanie

Oblicz użyteczności stanów dla poniższego świata za pomocą wybranego algorytmu (value iteration, policy iteration, modified policy iteration).



Zasady takie same jak w przykładzie. Przy czym $P(przód) = 0.5$, $P(lewo) = P(prawo) = 0.2$. $P(tył) = 0.1$.
$r = -0.04$ (kary)
$\gamma = 0.8$