Uczenie ze wzmocnieniem

(Reinforcement Learning)

Materiały dodatkowe

- Richard S. Sutton and Andrew G. Barto, *Reinforcement learning, an introduction,* second edition:

  http://incompleteideas.net/book/the-book-2nd.html

- David Silver, *UCL Course on RL:*

  http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html

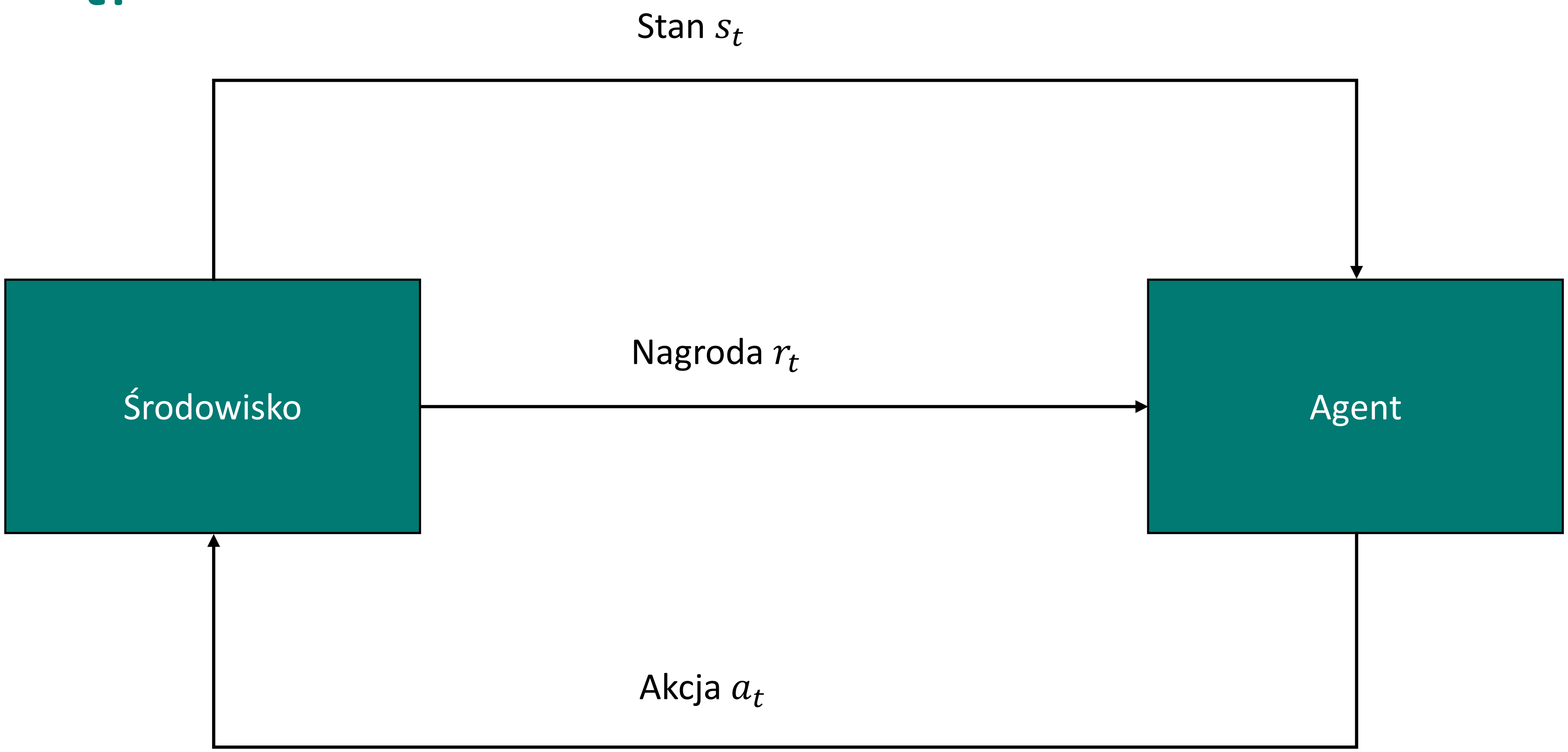
Wstęp

- 3 podstawowe podejścia do uczenia maszynowego:
  - Uczenie nadzorowane
  - Uczenie nienadzorowane
  - Uczenie ze wzmocnieniem

Wstęp

- 3 podstawowe podejścia do uczenia maszynowego:
  - Uczenie nadzorowane
  - Uczenie nienadzorowane
  - Uczenie ze wzmocnieniem
- Inspiracje:
  - Sieci neuronowe – budowa układu nerwowego żywych organizmów.
  - Uczenie ze wzmocnieniem – podejście behawioralne do uczenia.

Wstęp



Wstęp

Hipoteza nagrody - **celem agenta jest maksymalizacja wartości oczekiwanej jego skumulowanej użyteczności.**

Łańcuch Markowa

- Ciąg zdarzeń, w którym prawdopodobieństwo każdego zdarzenia zależy jedynie od wyniku poprzedniego

$$P(X_{n+1} \leqslant y | X_0, X_1, X_2, \ldots, X_n) = P(X_{n+1} \leqslant y | X_n)$$

Łańcuch Markowa

- Ciąg zdarzeń, w którym prawdopodobieństwo każdego zdarzenia zależy jedynie od wyniku poprzedniego

$$P(X_{n+1} \leqslant y | X_0, X_1, X_2, \ldots, X_n) = P(X_{n+1} \leqslant y | X_n)$$

- **Własność Markowa** – warunkowe rozkłady prawdopodobieństwa przyszłych stanów procesu są zdeterminowane wyłącznie przez jego bieżący stan, bez względu na przeszłość.

Łańcuch Markowa

- Znając $P\left(X_{n+1} = S_j \middle| X_n = S_i\right)$ możemy wyznaczyć rozkład warunkowy prawdopodobieństwa dotarcia do S_j pod warunkiem, że zaczęliśmy w stanie S_0, np.:

$$P\left(X_2 = S_j \middle| X_0 = S_0\right) = \sum_{k=1}^{p} P(X_1 = S_k | X_0 = S_0) P(X_2 = S_j | X_1 = S_k)$$

- W ogólnym przypadku:

$$P\left(X_{n+m} = S_j \middle| X_n = S_i\right) = \sum_{k=1}^{p} P(X_k = S_k | X_n = S_i) P(X_{n+m} = S_j | X_k = S_k)$$

Łańcuch Markowa

- Znając $P\left(X_{n+1} = S_j \middle| X_n = S_i\right)$ możemy wyznaczyć rozkład warunkowy prawdopodobieństwa dotarcia do S_j pod warunkiem, że zaczęliśmy w stanie S_0, np.:

$$P\left(X_2 = S_j \middle| X_0 = S_0\right) = \sum_{k=1}^{p} P(X_1 = S_k | X_0 = S_0) P(X_2 = S_j | X_1 = S_k)$$

- W ogólnym przypadku:

$$P\left(X_{n+m} = S_j \middle| X_n = S_i\right) = \sum_{k=1}^{p} P(X_k = S_k | X_n = S_i) P(X_{n+m} = S_j | X_k = S_k)$$

- Takie wyrażenie nazywamy równaniem Chapmana-Kołmogorowa

Łańcuch Markowa

- Macierz:

$$P^{(n)} = \begin{bmatrix} p_{11}^{(n)} & \cdots & p_{1n}^{(n)} \\ \vdots & \ddots & \vdots \\ p_{n1}^{(n)} & \cdots & p_{nn}^{(n)} \end{bmatrix}$$

nazywamy macierzą przejść.

- Gdy prawdopodobieństwa nie zależą od aktualnego stanu mówimy, że łańcuch jest jednorodny.

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix}$$

Łańcuch Markowa

- Gdy prawdopodobieństwa nie zależą od aktualnego stanu mówimy, że łańcuch jest jednorodny.

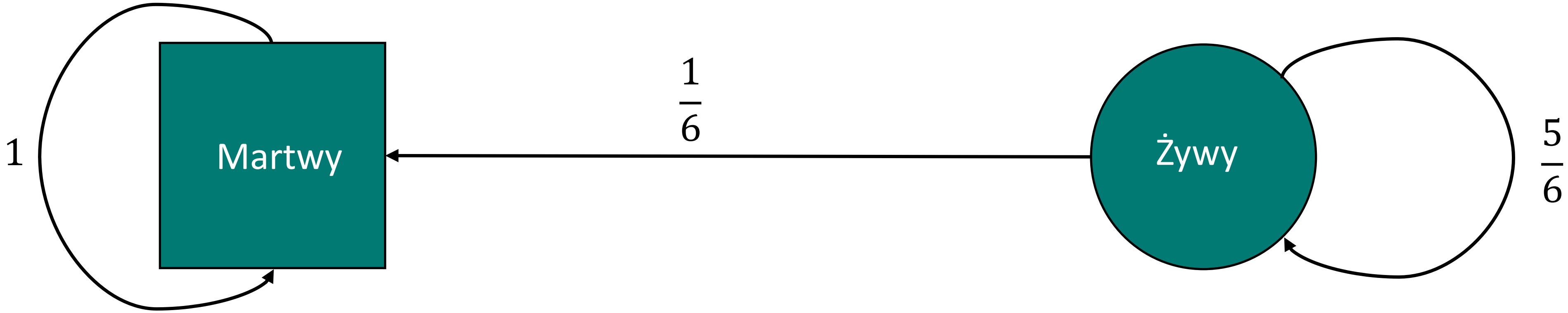
$$P = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix}$$

- Dla macierzy P równanie Chapmana-Kołmogorowa przyjmuje postać:

$$P^{m+n} = P^m P^n$$

Łańcuch Markowa

- Przykład – gra w Rosyjską ruletkę:



- Macierz przejścia:

$$P = \begin{bmatrix} 1 & 0 \\ \frac{1}{6} & \frac{5}{6} \end{bmatrix}$$

Łańcuch Markowa

- Klasyfikacja stanów:
  - Stan S_j jest osiągalny ze stanu S_i jeśli $P(S_j, S_i) \geq 0$
  - Stany S_j i S_i są skomunikowane jeśli są wzajemnie osiągalne
  - Gdy $\lim_{n \to +\infty} P(X_n = S_j | X_0 = S_0) = 0$ to stan S_j jest stanem chwilowym.
  - Gdy $\lim_{n \to +\infty} P(X_n = S_j | X_0 = S_0) = 1$ to stan S_j jest stanem pochłaniającym.
  - Gdy $\lim_{n \to +\infty} P(X_n = S_j | X_0 = S_0) \in (0,1)$ to stan S_j jest stanem powracającym.

Łańcuch Markowa

- Dla łańcucha Markowa składającego się tylko ze stanów chwilowych i powracających możemy wyznaczyć rozkład stacjonarny:

$$\lim_{n \to +\infty} P^n = \begin{bmatrix} p_1 & \cdots & p_n \\ \vdots & \ddots & \vdots \\ p_1 & \cdots & p_n \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} p_1 & \cdots & p_n \end{bmatrix}$$

Przykład

Monopoly

- Monopoly to jedna z najpopularniejszych gier planszowych w historii; jej historia sięga lat 30 XX wieku.
- Spróbujmy przedstawić ją jako łańcuch Markowa.
- Badamy tylko poruszanie się, na razie nie uwzględniamy decyzji graczy.

Przykład

Monopoly

HASBRO

- Gracze rzucają dwiema sześciennymi kostkami i przesuwa swój pionek o ilość pół, którą wyrzucił kostkami.
- Dla uproszczenia pomińmy zasady związane z dubletami (taką samą liczbę oczek na obu kostkach) - wyraźnie uprości to grę (dlaczego?).

Przykład

Monopoly

- Dodatkowo interesują nas jedynie karty typu „Kasa Społeczna”.
  - Takich kart jest **16**; 1 cofa gracza na **początek**, 1 wysyła go do **więzienia**, pozostałe nie mają wpływu na jego pozycję.
  - Karty typu „Szansa” nie zmieniają wyników a byłyby uciążliwe do zakodowania ;)

Proces Markowa z nagrodami

- Proces Markowa z nagrodami (*Markov reward process* – MRP) definiujemy jako następującą krotkę $(\boldsymbol{S}, \boldsymbol{P}, \boldsymbol{R}, \boldsymbol{\beta})$:
  - $\boldsymbol{S}$ **–** jest skończonym zbiorem stanów.
  - $\boldsymbol{P}(\mathbf{s}, \mathbf{s}')$**–** Prawdopodobieństwo znalezienia się w stanie $\mathbf{s}'$ w czasie $t+1$ pod warunkiem znajdowania się w stanie s w czasie t.
  - $\boldsymbol{R}(\mathbf{s}, \mathbf{s}')$ **–** Nagroda za przejście ze stanu s do stanu $\mathbf{s}'$ w czasie t.
  - $\boldsymbol{\beta}$ **–** współczynnik dyskontujący $\beta \in [0,1)$

Proces Markowa z nagrodami

- Skumulowana przyszła nagroda:

$$R_t = r_{t+1} + \beta r_{t+2} + \cdots + \beta^{\mathrm{T}-1} r_{\mathrm{T}-t} = \sum_{k=1}^{\mathrm{T-t}} \beta^{k-1} r_{t+k}$$

- **Funkcję wartości** $V(s)$ definiujemy jako oczekiwaną wypłatę zaczynając od stanu s

$$V(s) = E[R_t | S_t = s]$$

Równanie Bellmana

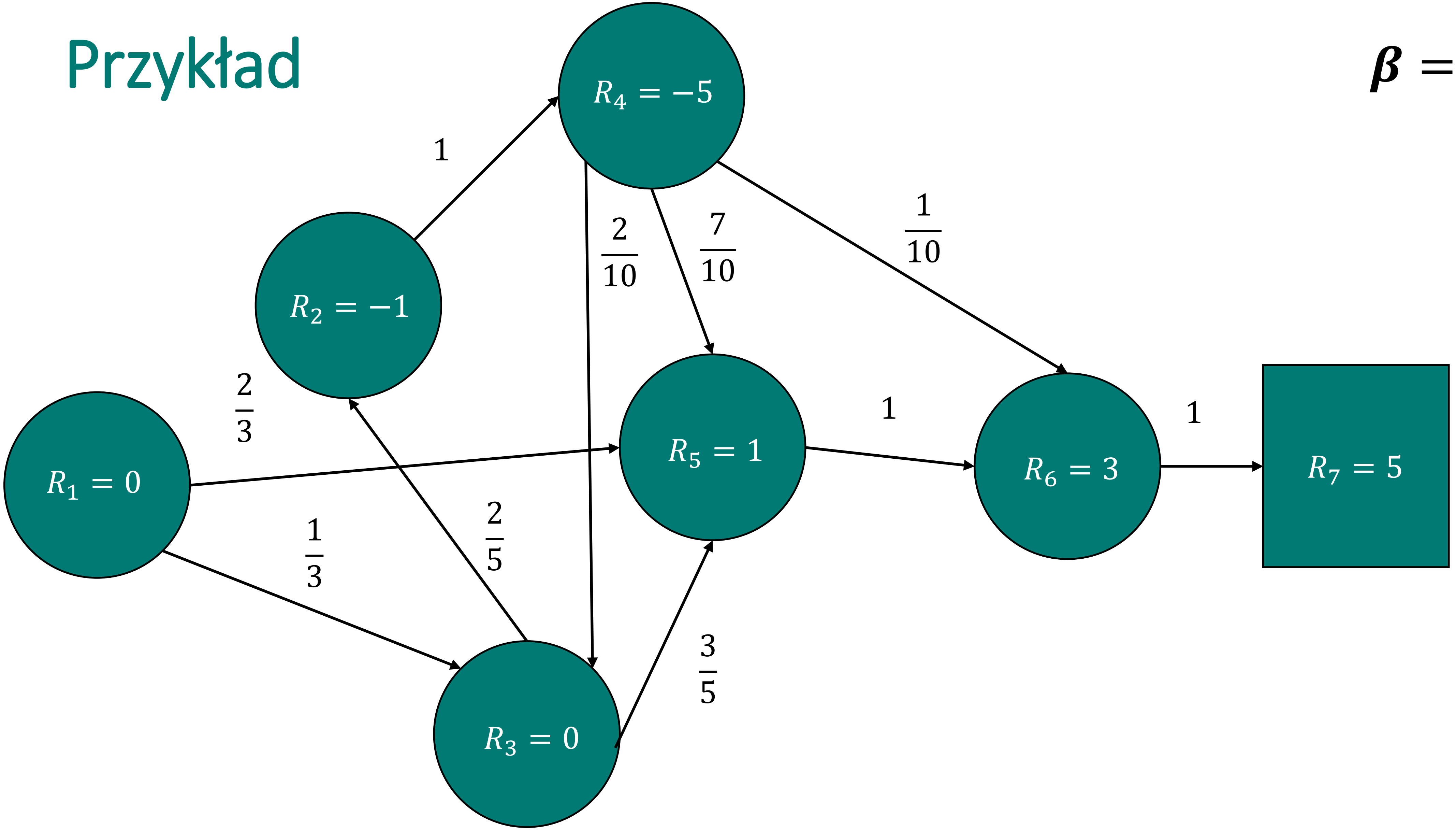
- Oba równania można zdekomponować do postaci natychmiastowej nagrody i zdyskontowanej przyszłej wartości:

$$V(s) = E[R_t | S_t = s] = E[r_{t+1} + \beta V(S_{t+1}) | S_t = s]$$

Taka postać nosi nazwę **Równania Bellmana**

Przykład

$\boldsymbol{\beta = 1}$

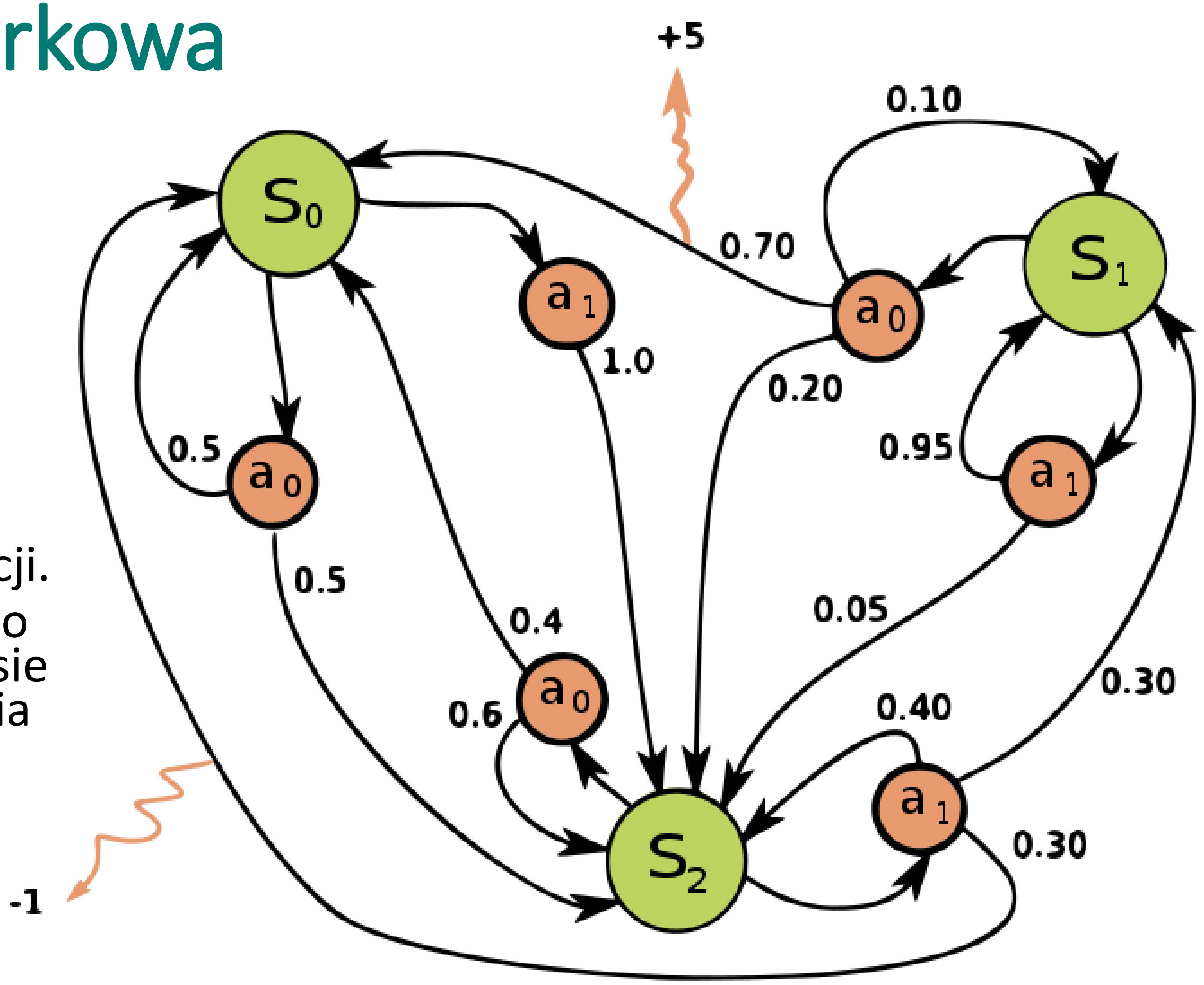


Proces decyzyjny Markowa

- Proces decyzyjny Markowa (*Markov decision process* – MDP) definiujemy jako następującą krotkę $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{R}, \boldsymbol{\beta})$:
  - $\boldsymbol{S}$ – jest skończonym zbiorem stanów.
  - $\boldsymbol{A}$ – jest skończonym zbiorem akcji.
  - $\boldsymbol{P}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$– Prawdopodobieństwo znalezienia się w stanie $\mathbf{s}'$ w czasie $t+1$ w wyniku podjęcia działania a w stanie s w czasie t.
  - $\boldsymbol{R}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ – Nagroda za przejście ze stanu s do stanu $\mathbf{s}'$ wyniku podjęcia działania a w czasie t.
  - $\boldsymbol{\beta}$ – współczynnik dyskontujący $\beta \in [0,1)$

Proces decyzyjny Markowa

- Proces decyzyjny Markowa (*Markov decision process* – MDP) definiujemy jako następującą krotkę $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{R}, \boldsymbol{\beta})$:
  - $\boldsymbol{S}$ – jest skończonym zbiorem stanów.
  - $\boldsymbol{A}$ – jest skończonym zbiorem akcji.
  - $\boldsymbol{P}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$– Prawdopodobieństwo znalezienia się w stanie $\mathbf{s}'$ w czasie $t + 1$ w wyniku podjęcia działania a w stanie s w czasie t.
  - $\mathbf{R}(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ – Nagroda za przejście ze stanu s do stanu $\mathbf{s}'$ wyniku podjęcia działania a w czasie t.
  - $\boldsymbol{\beta}$ – współczynnik dyskontujący $\beta \in [0,1)$



Proces decyzyjny Markowa

- **Strategię** definiujemy jako:

$$\pi \colon S \to A$$
$$\pi(a|s) = P[A_t = a|S_t = s]$$

- Strategia w pełni opisuje zachowania agenta.
- Strategie zależą jedynie od aktualnego stanu MDP a nie od historii.
- Są *stacjonarne* (nie zależą od czasu).

Proces decyzyjny Markowa

- **Funkcję wartości stanu** *(state-value function)* $V_\pi(s)$ definiujemy jako oczekiwaną wypłatę zaczynając od stanu s i podążając za strategią π:

$$V_\pi(s) = E_\pi[R_t | S_t = s]$$

- **Funkcję wartości akcji** *(action-value function)* $q_\pi(s, a)$ definiujemy jako oczekiwaną wypłatę zaczynając od stanu s, podejmując akcję a i podążając za strategią π:

$$q_\pi(s, a) = E_\pi[R_t | S_t = s \mid A_t = a]$$

Równanie Bellmana

- Oba równania można zdekomponować do postaci natychmiastowej nagrody i zdyskontowanej przyszłej wartości:

$$V_\pi(s) = E_\pi[R_t|S_t = s] = E_\pi[r_{t+1} + \beta V_\pi(S_{t+1})|S_t = s]$$

$$q_\pi(s,a) = E_\pi[R_t|S_t = s \mid A_t = a] = E_\pi[r_{t+1} + \beta q_\pi(S_{t+1}, A_{t+1})|S_t = s \mid A_t = a]$$

Taka postać nosi nazwę **Równania Bellmana**

Równanie Bellmana

- Upraszczając możemy zapisać oba równania w formie:
  - $V_\pi(s) = R(s) + \beta P(s, \pi(s), s')V_\pi(s')$
  - $q_\pi(s, \pi(s)) = R(s) + \beta P(s, \pi(s), s')q_\pi(s', \pi(s'))$

Proces decyzyjny Markowa

- **Optymalna funkcja wartości stanu** V_*:

$$V_*(s) = \max_{\pi} V_\pi(s, a)$$

- **Optymalna funkcja wartości akcji** $q_*(s, a)$:

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

Równanie Bellmana

- Dla każdego problemu, który można zdefiniować jako proces decyzyjny Markowa:
  - istnieje taka optymalna, deterministyczna strategia π_*, która jest nie gorsza od pozostałych dostępnych $\pi_*(s) \geq \pi(s), \ \forall\pi \ \forall s \in S$
  - Optymalna strategia π_* osiąga optymalną wartość funkcji wartości $V_{\pi_*}(s) = V_*(s)$.
  - Optymalna strategia π_* osiąga optymalną wartość funkcji wartości akcji $q_{\pi_*}(s,a) = q_*(s,a)$.

Równanie Bellmana

- Definiujemy ją jako:

$$\pi_*(s|a) = \begin{cases} 1 \ \ gdy\ a = \underset{a \in A}{\operatorname{argmax}}\, q_*(s,a) \\ 0\ w\ przeciwnym\ wypadku \end{cases}$$

- Aby ją znaleźć musimy znaleźć maksymalizować po $q_*(s,a)$.
- Gdy znamy $q_*(s,a)$ znalezienie optymalnej strategii jest trywialne.

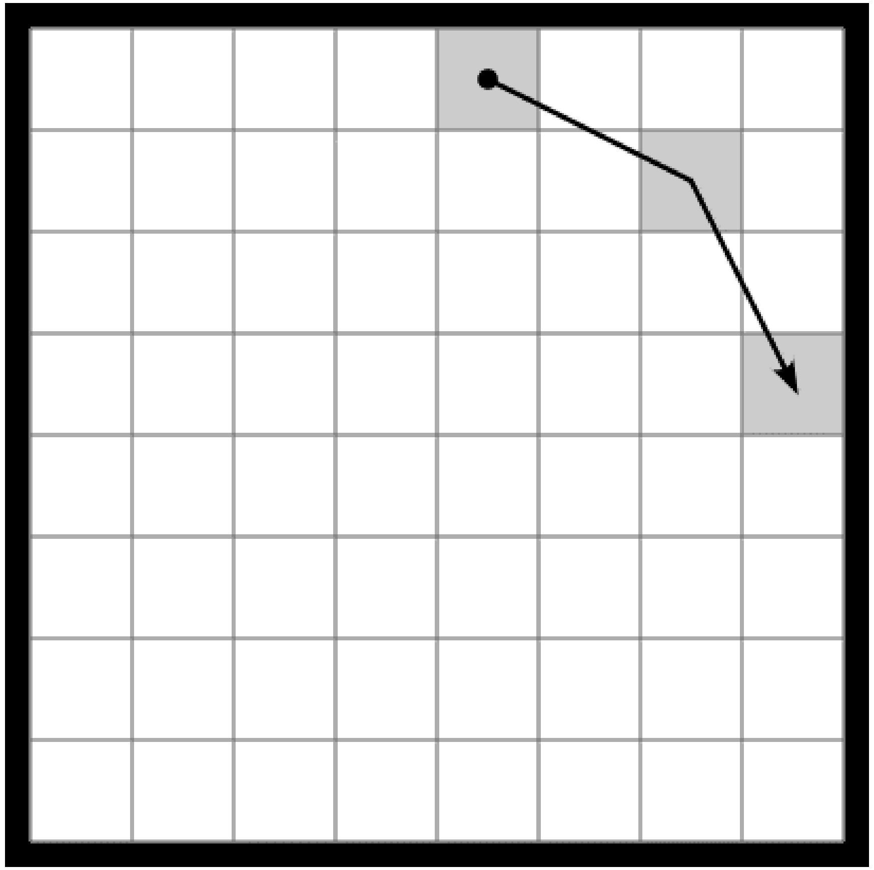
Wyszukiwanie wyczerpujące

Exhaustive search

- Równania Bellmana można rozwiązać analitycznie.
- Złożoność obliczeniowa sięga $\mathrm{O}(\mathrm{n}^3)$ gdzie n oznacza liczbę stanów.
- W przypadku rozwiązywania dużych problemów istnieje wiele metod iteracyjnych:
  - Exhaustive search
  - Programowanie dynamiczne
  - Metody Monte Carlo
  - Temporal Difference Learning

Przykład

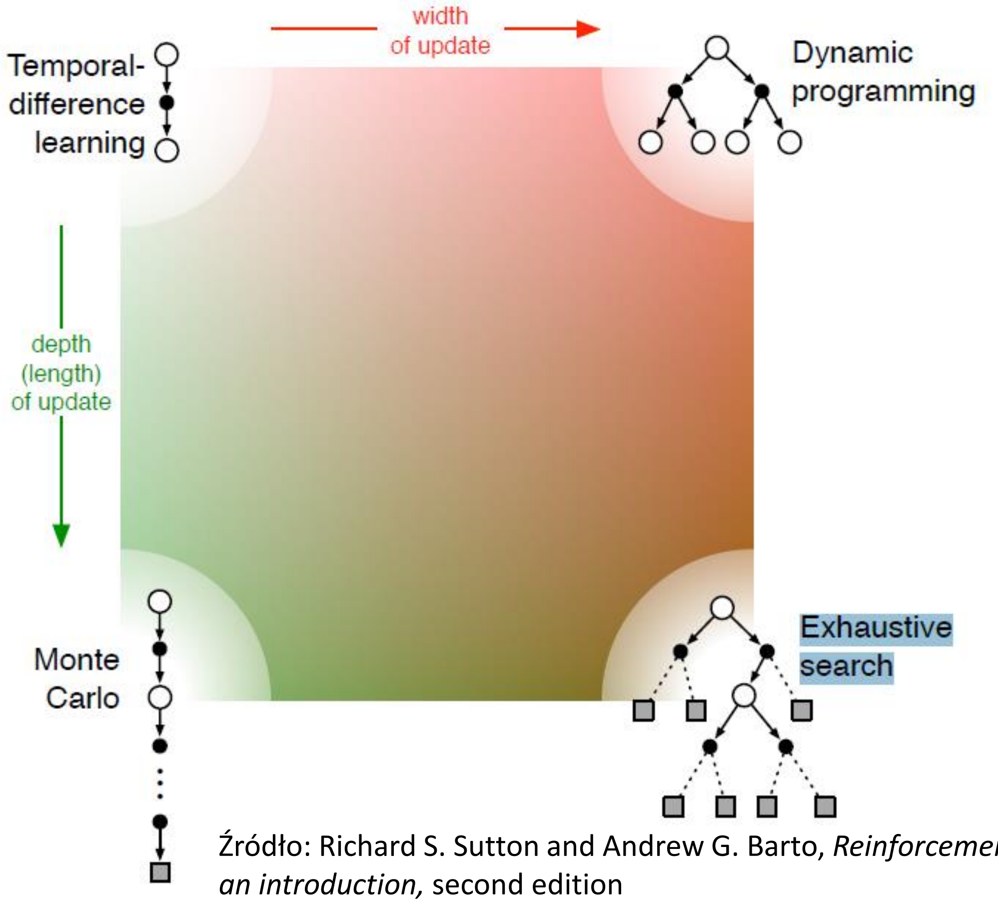
Skoczek na szachownicy



- Celem jest obejście skoczkiem wszystkich pól planszy tak, żeby na każdym polu stanąć raz i tylko raz.

Exhaustive search

- Np. backtracking
- Metody nieefektywne – wymagają przechowywania dużej ilości informacji:



Źródło: Richard S. Sutton and Andrew G. Barto, *Reinforcement learning, an introduction,* second edition