Riešenie MDP

Juraj Gazda

iislab.kpi.fei.tuke.sk

Technical University of Košice
Department of Computers and Informatics

Odmeny a epizódy

Reward vs return (odmena vs úžitok)
Agent: entita maximalizujúca očakávanú kumulatívnu odmenu:

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

Diskontovanie:

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

Základné pravidlo RL (guess from the guess in general):

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

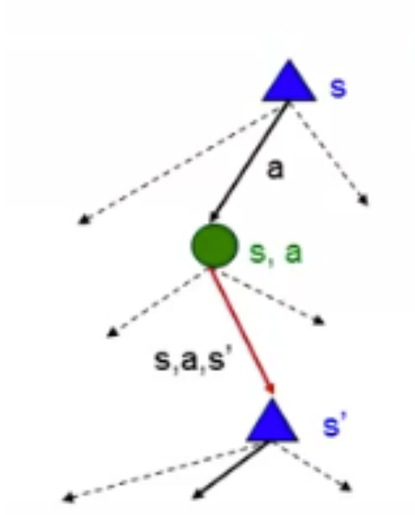
Hodnota stavu, q stav

Hodnota (úžitok) stavu S

$v_*(s)$ = očakávaný úžitok zo stavu *s* nasledujúc opt. strat.profil

Hodnota (úžitok) stavu *q(s,a)*

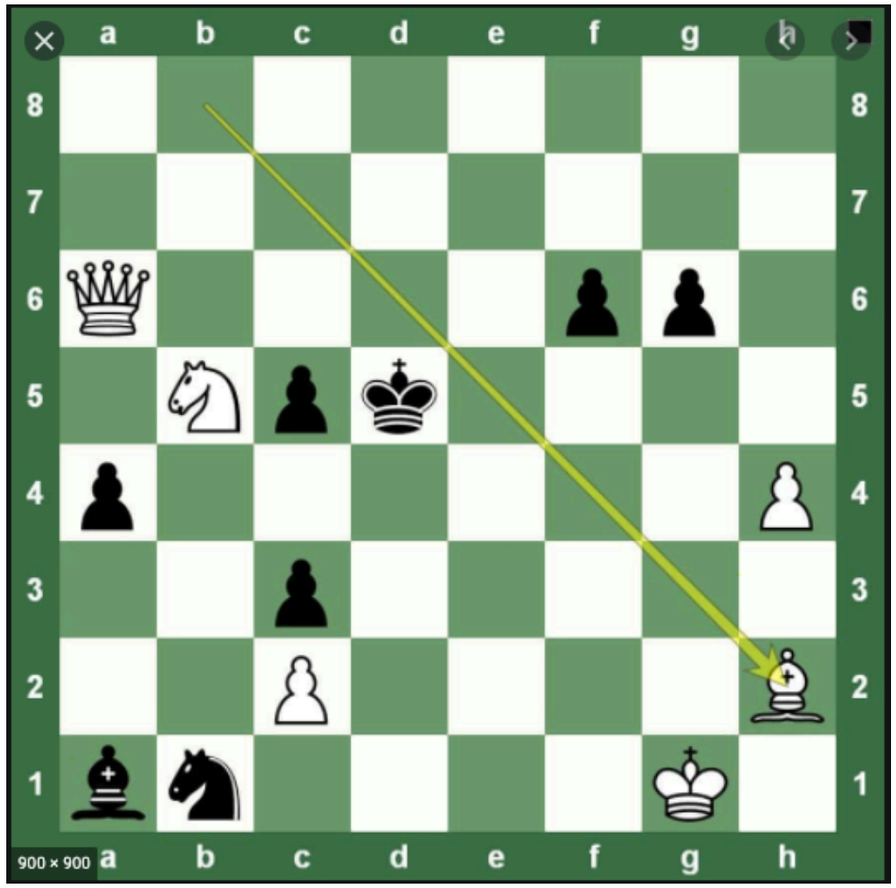
$Q_*(s,a)$ = očakávaný úžitok zo stavu *s* po vykonaní akcie *a* nasledujúc opt. strat.profil



Optimálny strategický profil $\pi^*(s)$

$\pi_*(s)$ = optimálna akcia zo stavu *s*, t.j. $V_*(s) \leftarrow \max Q_*(s,a)$

Hodnota stavu, q stav



Výhra 100 bodov, prehra 100 bodov

π náhodný strategický profil

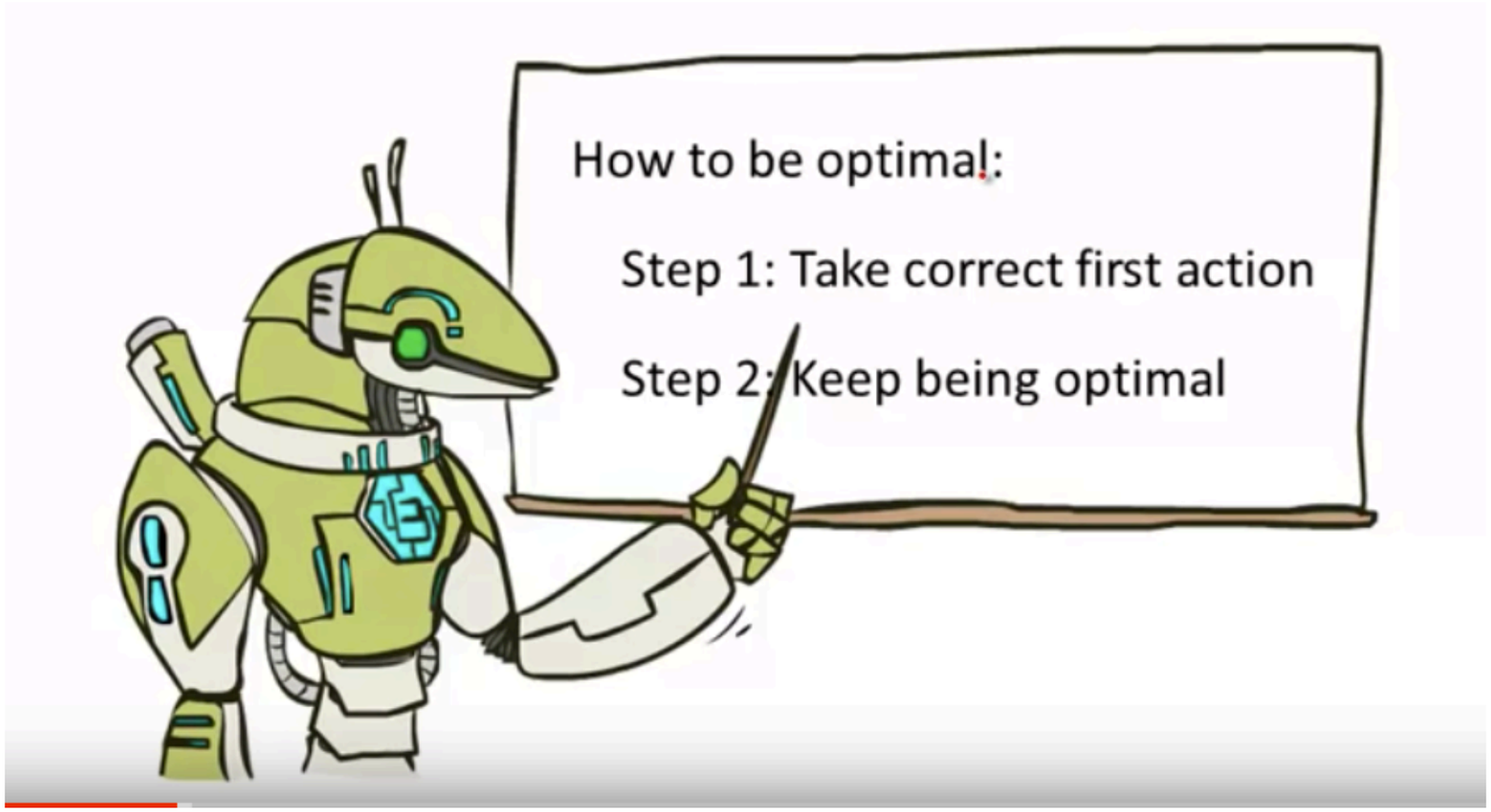
Biely hráč $v^*(s) = 99$

Čierny hráč $v^*(s) = 0{,}01$

Biely hráč $v_\pi(s) = 5$

Čierny hráč $v_{\pi'}(s) = 2$

Hodnota stavu, q stav



Hodnota stavu, q stav

Hodnota stavu $v_\pi(s)$ pre ľubovoľný strateg. profil π

$$v_\pi(s) \doteq \mathbb{E}_\pi\left[G_t \,|\, S_t = s\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,|\, S_t = s\right], \forall s \in \mathcal{S}$$

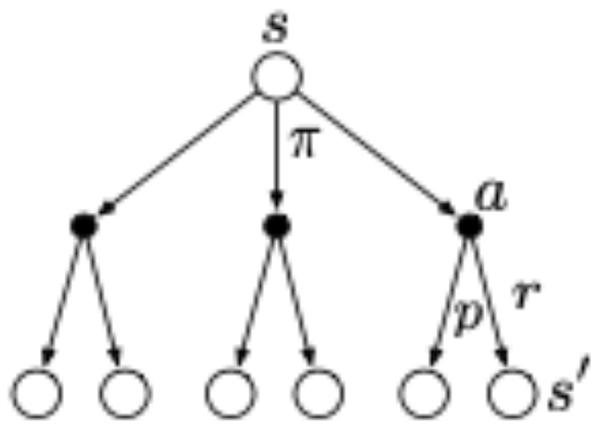
q hodnota pre stav *s* a akciu *a*, *q(s,a)* pre strat. profil π

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

Rekurzívna implementácia výpočtu *v(s)*

Bellmanove rovnice (Bellman equations)

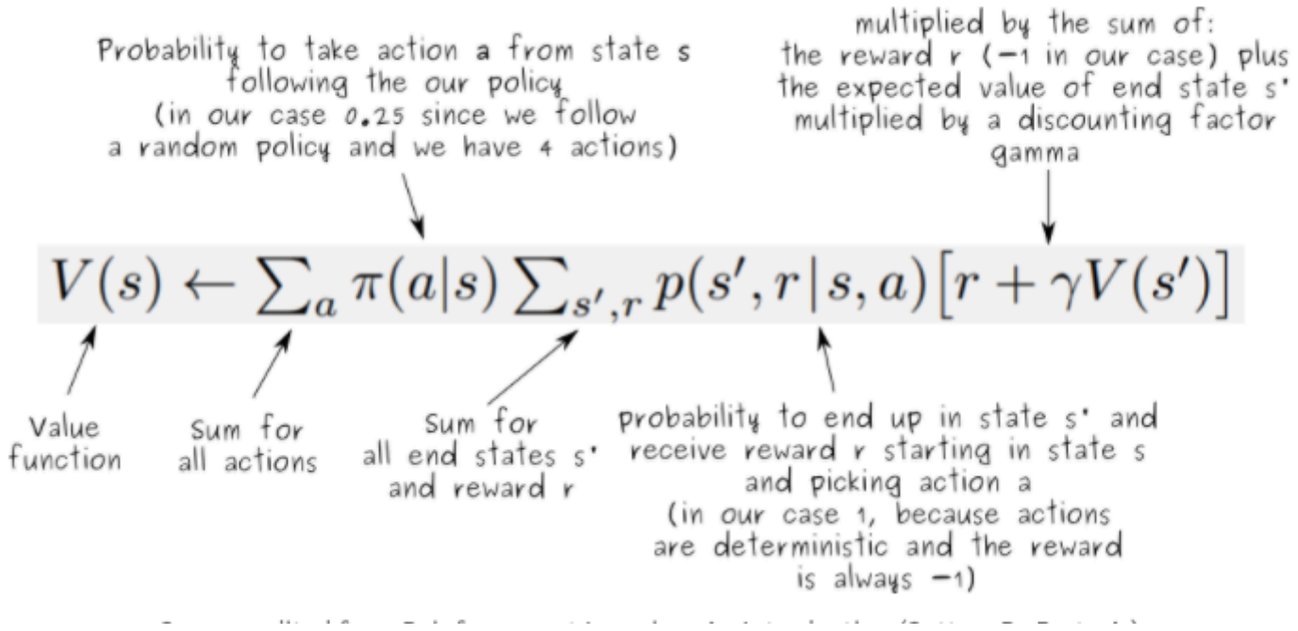
$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi\left[G_t \,|\, S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \,|\, S_t = s\right] \\
&= \sum_a \pi(a\,|\,s) \sum_{s'} \sum_r p\left(s', r \,|\, s, a\right)\left[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \,|\, S_{t+1} = s'\right]\right]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



Uvažujme *N* stavov, potom riešenie môžeme získať riešením *N* lineárnych rovníc

Nie je riešiteľné, potrebujeme vedieť pravú stranu rovnice. Využijeme aproximatívny prístup

Rekurzívna implementácia výpočtu *v(s)*



Policy evaluation (predikcia kvality str. profilu)

Motivácia: určenie kvality (hodnoty, resp. využitím hodnotovej funkcie $v_\pi(s), \forall s$) strategického profilu π

V literatúre známe ako *prediction problem* - príklad **policy evaluation**

Control problem - snaha iteratívnym spôsobom zlepšiť strategický profil
- príklad: **policy improvement, value iteration**

Policy evaluation (predikcia kvality str. profilu)

Úloha 1: Výpočet hodnôt stavov pre ľubovoľný strategický profil π

V literatúre známe ako POLICY EVALUATION, ktorý rieši tzv. prediction problém

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

Iteratívny prístup: Na začiatku algoritmu priraďme stavom náhodné hodnoty stavov a v ďalších iteráciách hodnoty stavov obnovujme podľa

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

Policy evaluation (predikcia kvality str. profilu)

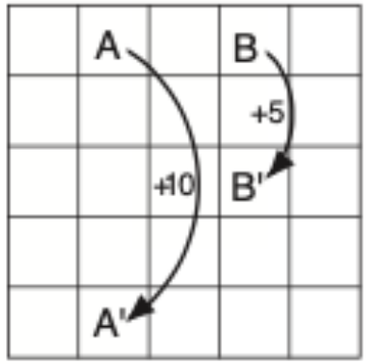
Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Príklad 3.2 (+ implementácia v Pythone)

Example 3.5: Gridworld Figure 3.2 (left) shows a rectangular gridworld representation of a simple finite MDP. The cells of the grid correspond to the states of the environment. At each cell, four actions are possible: `north`, `south`, `east`, and `west`, which deterministically cause the agent to move one cell in the respective direction on the grid. Actions that would take the agent off the grid leave its location unchanged, but also result in a reward of -1. Other actions result in a reward of 0, except those that move the agent out of the special states A and B. From state A, all four actions yield a reward of $+10$ and take the agent to A'. From state B, all actions yield a reward of $+5$ and take the agent to B'.



Actions

3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

Bellman optimality equation (control problem)

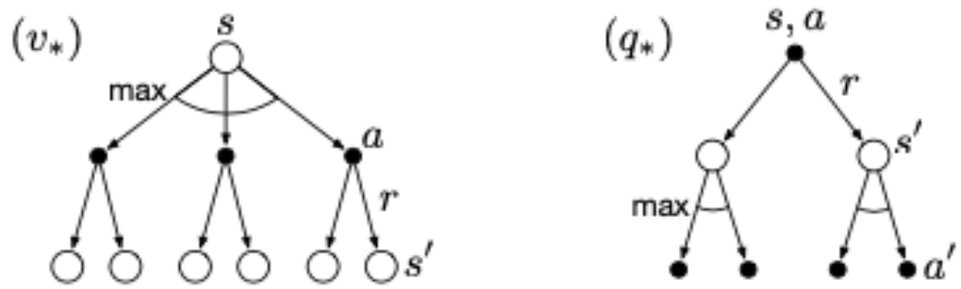
Tvrdenie: Hodnotu stavu $v_{\pi*}(s)$ pri opt. strategickom profile π_* je je možné určiť ako akciu *a* prinášajúcu maximálny očakávaný úžitok danom stave *s*

$$v_*(s) = \max_a q_*(s,a)$$

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

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

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



Uvažujme *N* stavov, potom riešenie môžeme získať riešením *N* NElineárnych rovníc

$$v_*(s) = \max_{a \in \mathcal{A}(s)} q_*(s,a)$$

$$= \max_a \mathbb{E}_{\pi_*}\left[G_t | S_t = s, A_t = a\right]$$

$$= \max_a \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a\right]$$

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

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

Policy iteration (iterácia strat. profilu)

$$\pi_0 \xrightarrow{\text{E}} v_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} v_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} v_*,$$

Control problem

E - Policy evaluation

I - Policy improvement

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

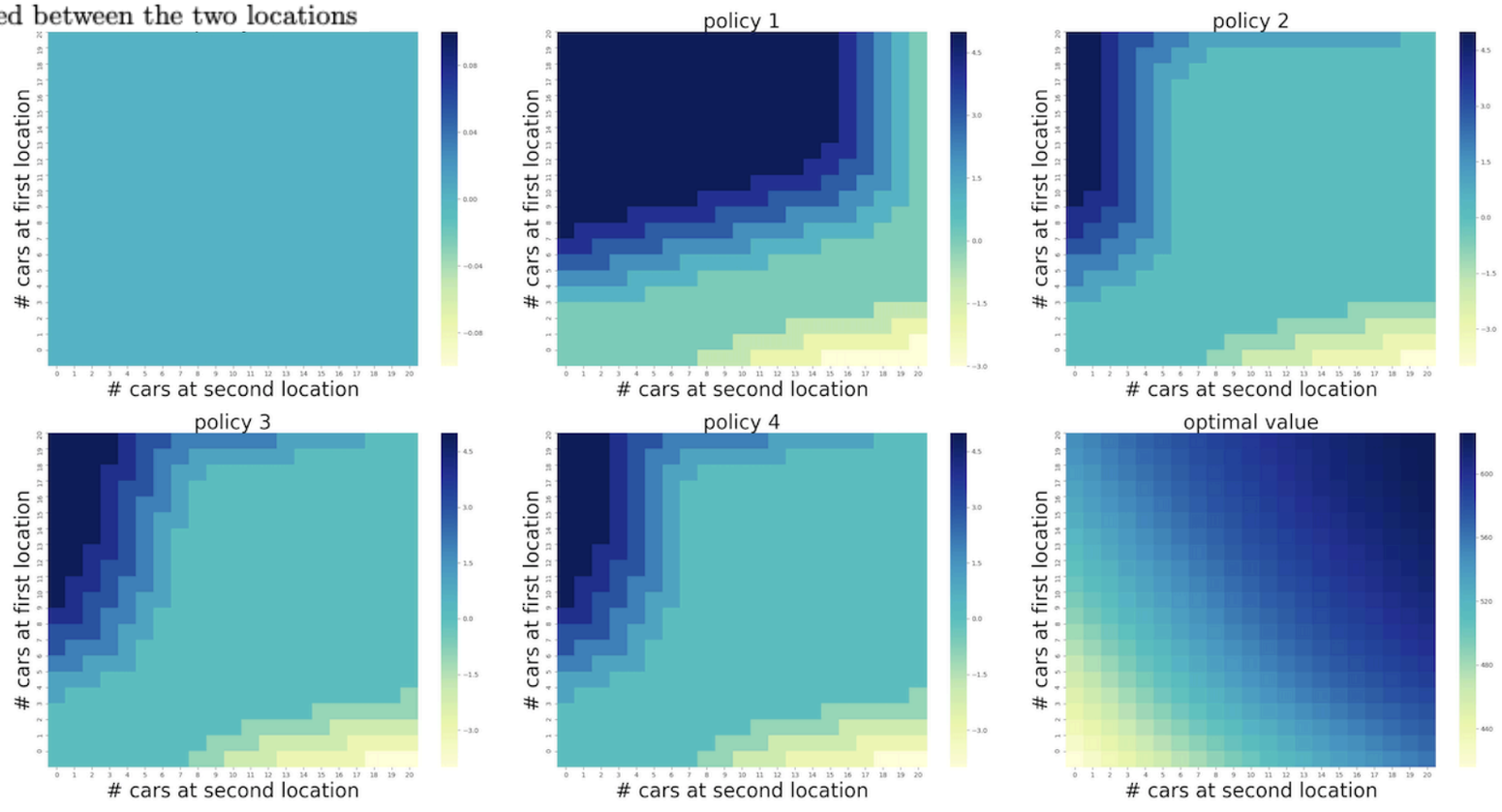
1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s)) \left[r + \gamma V(s') \right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma V(s') \right]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Príklad 3.2 (+ implementácia v Pythone)

Example 4.2: Jack's Car Rental Jack manages two locations for a nationwide car rental company. Each day, some number of customers arrive at each location to rent cars. If Jack has a car available, he rents it out and is credited \$10 by the national company. If he is out of cars at that location, then the business is lost. Cars become available for renting the day after they are returned. To help ensure that cars are available where they are needed, Jack can move them between the two locations overnight, at a cost of \$2 per car moved. We assume that the number of cars requested and returned at each location are Poisson random variables, meaning that the probability that the number is n is $\frac{\lambda^n}{n!}e^{-\lambda}$, where λ is the expected number. Suppose λ is 3 and 4 for rental requests at the first and second locations and 3 and 2 for returns. To simplify the problem slightly, we assume that there can be no more than 20 cars at each location (any additional cars are returned to the nationwide company, and thus disappear from the problem) and a maximum of five cars can be moved from one location to the other in one night. We take the discount rate to be $\gamma = 0.9$ and formulate this as a continuing finite MDP, where the time steps are days, the state is the number of cars at each location at the end of the day, and the actions are the net numbers of cars moved between the two locations



Value iteration (hodnotová iterácia)

Motivácia: eliminovať nutnosť volania policy evaluation pri hľadaní optim. strateg. profilu.
Iteratívne zvyšovanie hodnôt jednotlivých stavov je možné vykonať bez nutnej evaluácie strategického profilu, priamo v jednom kroku. Garancia konvergencie je zaručená (neuvádzame ju)

Control problem

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

Value iteration (hodnotová iterácia)

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\mathit{terminal}) = 0$

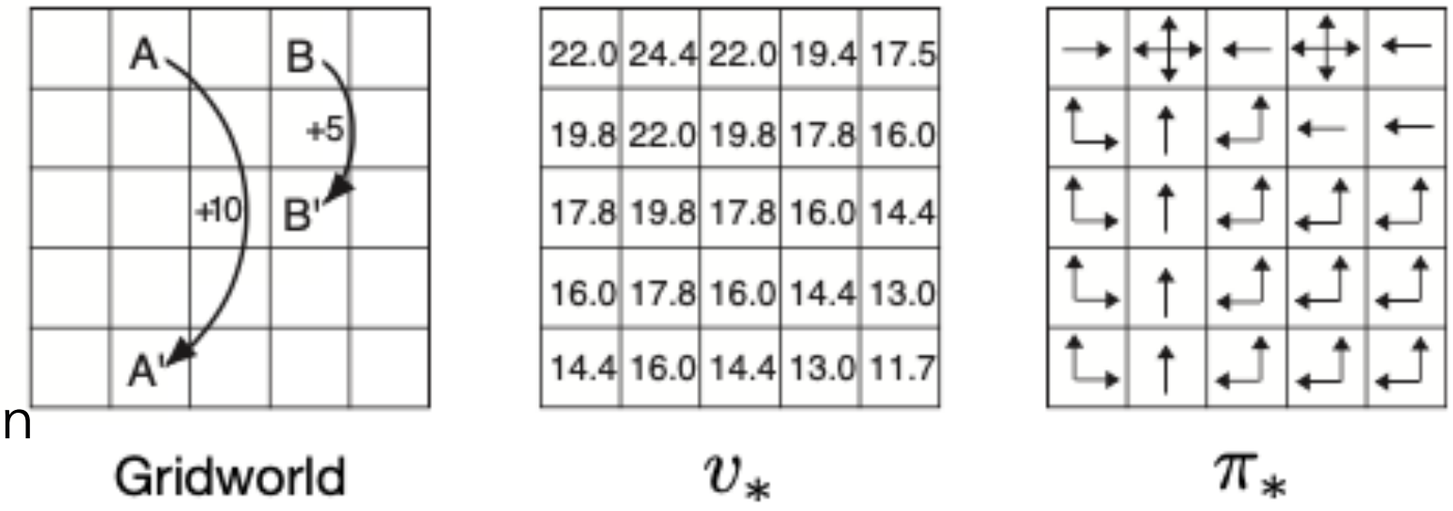
Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
  $\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$

Control problem

Príklad 3.8 (+ implementácia v Pythone)

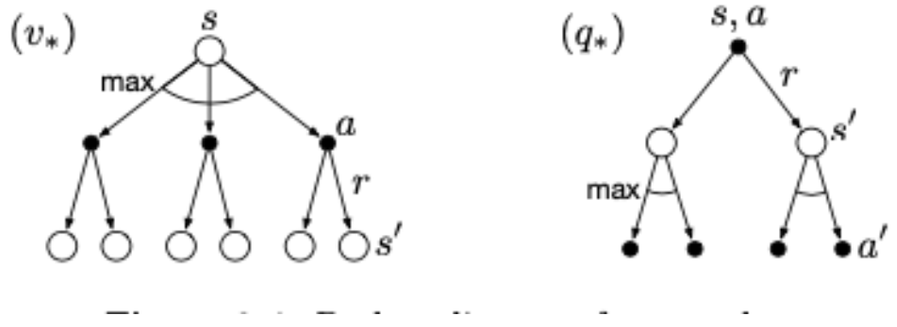
Example 3.8: Solving the Gridworld Suppose we solve the Bellman equation for v_* for the simple grid task introduced in Example 3.5 and shown again in Figure 3.5 (left). Recall that state A is followed by a reward of +10 and transition to state A′, while state B is followed by a reward of +5 and transition to state B′. Figure 3.5 (middle) shows the optimal value function, and Figure 3.5 (right) shows the corresponding optimal policies. Where there are multiple arrows in a cell, all of the corresponding actions are optimal.



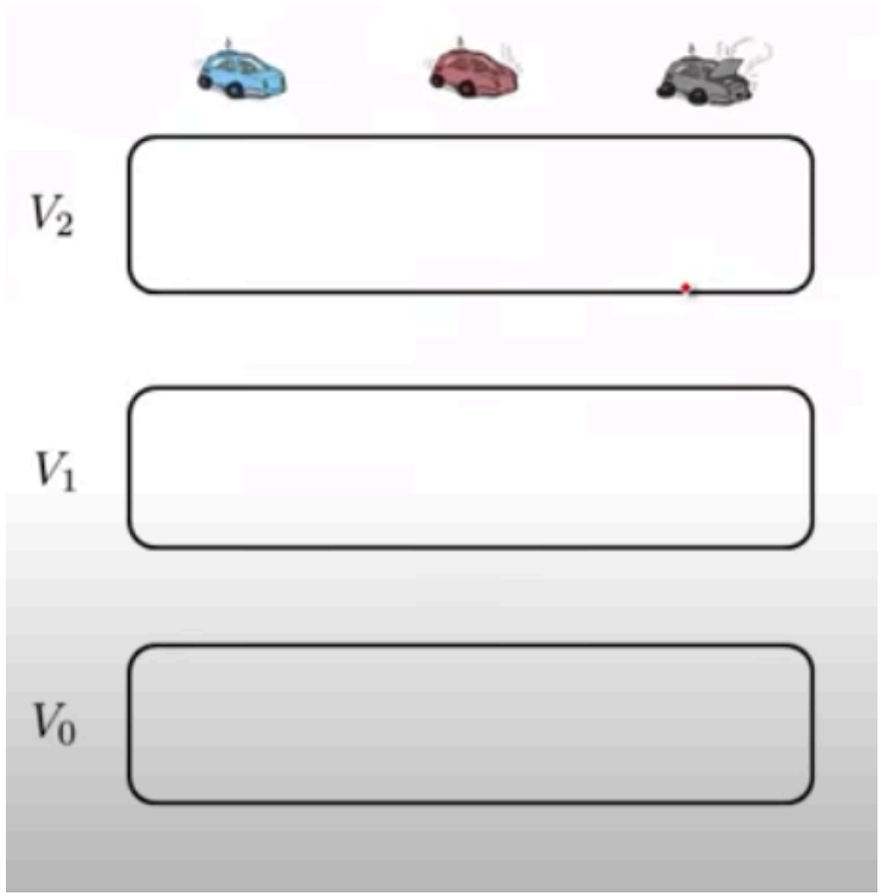
Gridworld v_* π_*

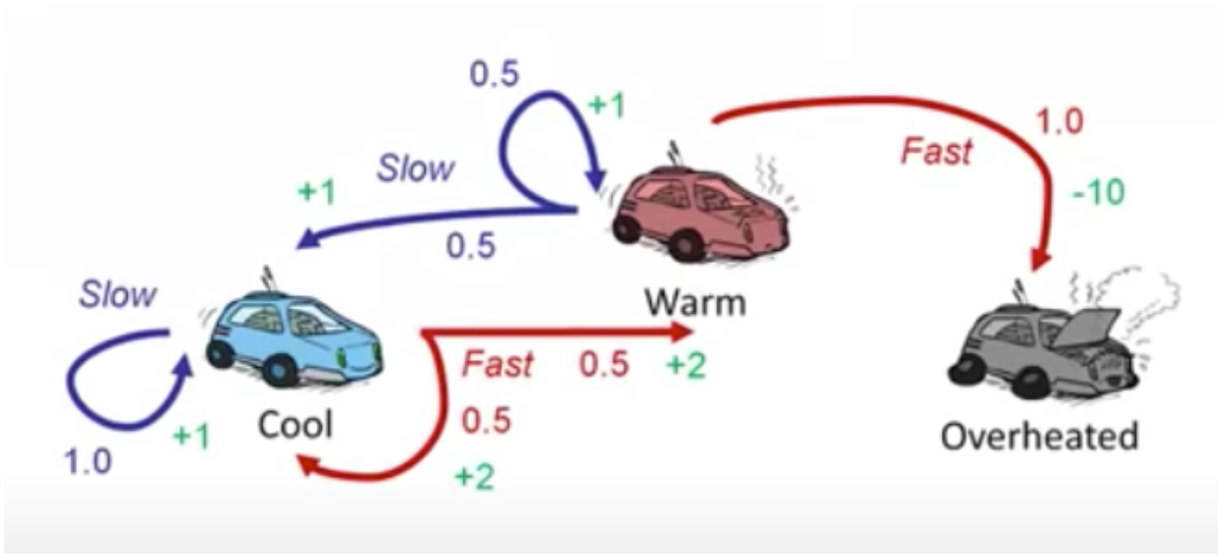
Policy extraction

$$\pi_*(s) = \arg\max_a \sum_{s'} p(s', r \mid s, a)[R(s, a, s') + \gamma V_*(s')]$$



Value iteration (hodnotová iterácia)

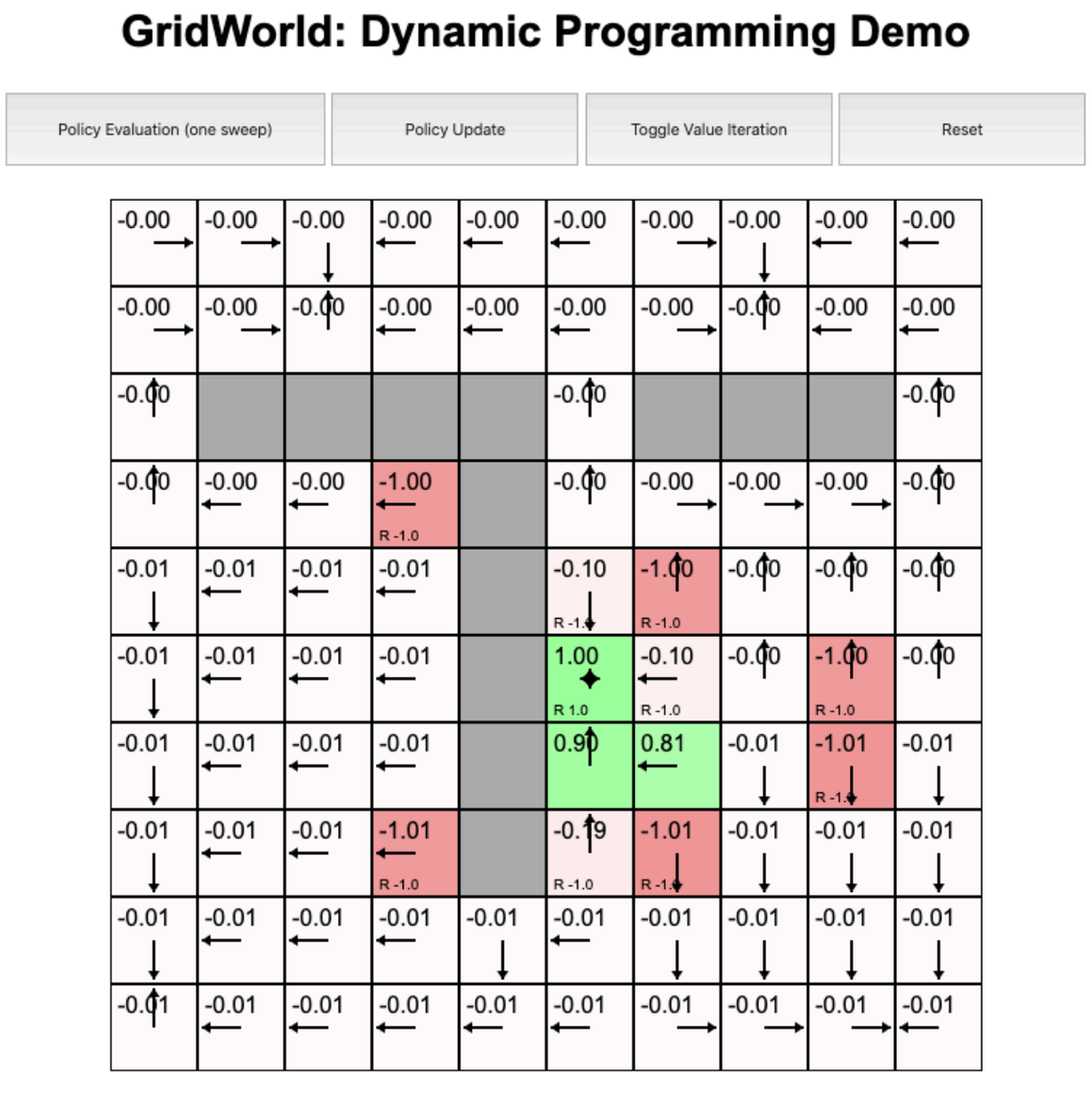




$$v_{k+1}(s) = \max_a \sum_{s',r} p\left(s', r \,|\, s, a\right) \left[r + \gamma v_k(s')\right]$$

Value iteration (hodnotová iterácia)

https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_dp.html



Zhrnutie

- ☆ Naučili sme sa:
  - Výpočet hodnotovej funkcie pre daný strategický profil (**policy evaluation**)
  - Nájdenie optimálneho strategického profilu iteratívnym zlepšením hodnotovej funkcie stavov (**value iteration)**, resp. implementovaním **policy evaluation** v kombinácií s **policy extraction**