Probabilistic Decision Making

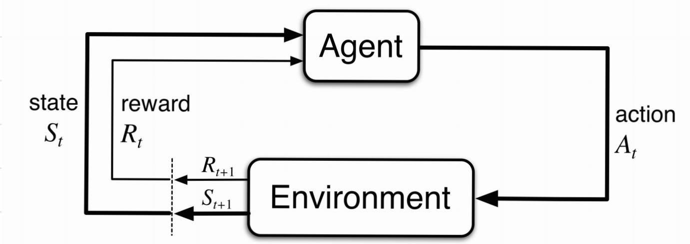
Lecture 15

Reinforcement Learning 2

Robert Peharz

Institute of Machine Learning and Neural Computation
Graz University of Technology
Winter Term 2025/26

Recap



Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, p)$

state space $\mathcal{S}$, action space $\mathcal{A}$, dynamics $p(s', r \mid s, a)$

Policy $\pi(a \mid s)$

Discounted Return

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad , \quad \gamma \in [0,1]$$

Value Function

$$v_\pi(s) := \mathbb{E}_\pi [G_t \mid S_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s \right]$$

Recap

Image: wikipedia.org

Bellman Expectation Equation

$$\underline{v_\pi(s)} = \mathbb{E}_{A_t, R_{t+1}, S_{t+1}}\left[R_{t+1} + \gamma \underline{v_\pi(S_{t+1})} \mid S_t = s \right]$$

$$\underline{v_\pi(s) = r(s) + \gamma \sum_{s'} p(s'|s)\, v_\pi(s')} \qquad \forall s \in \mathcal{S}$$

$$\boxed{p(s'|s) = \sum_{r,a} p(s', r | s, a)\, \pi(a|s)}$$

Matrix-Vector form:

$$\underline{v} = \begin{pmatrix} v_\pi(s_1) \\ v_\pi(s_2) \\ \vdots \\ v_\pi(s_N) \end{pmatrix} \qquad \underline{r} = \begin{pmatrix} r(s_1) \\ r(s_2) \\ \vdots \\ r(s_N) \end{pmatrix} \qquad P = \begin{pmatrix} p(s_1|s_1) & \dots & p(s_N|s_1) \\ \vdots & \ddots & \vdots \\ p(s_1|s_N) & \dots & p(s_N|s_N) \end{pmatrix}$$

$$\boxed{\underline{v} = \underline{r} + \gamma P \underline{v}} \rightarrow \text{Analytic solution: } \underline{v_\pi} = (I - \gamma P)^{-1}\, \underline{r}$$

Recap

Iterative Policy Evaluation

Bellman equation
(interpreted as update rule)

Iterative Policy Evaluation

- initialize $v(s)$ arbitrarily (e.g. all 0)
- repeat
  - for $s \in \mathcal{S}$ do $v_{new}(s) \leftarrow r(s) + \gamma \sum_{s'} p(s'|s)\, v(s')$
  - if $\forall s: v_{new}(s) \approx v(s) \longrightarrow$ break
  - $v \leftarrow v_{new}$

Random walk policy $\pi(a|s) = 0.25$ $(\leftarrow, \rightarrow, \uparrow, \downarrow)$

k=0

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

k=1

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

k=2

0	-1.7	-2	-2
-1.7	-2	-2	-2
-2	-2	-2	-1.7
-2	-2	-1.7	0

. . .

K=10

0	-6.1	-8.3	-8.9
-6.1	-7.7	-8.4	-8.3
-8.3	-8.4	-7.7	-6.1
-8.9	-8.3	-6.1	0

. . .

K=199

0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

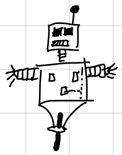
K=200

0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

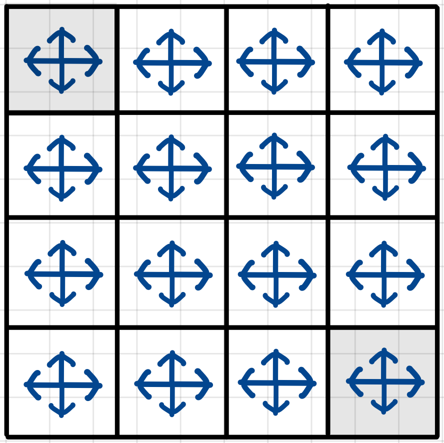
$\approx v_\pi$

How to Improve Policies?

Policy:



$\pi(a|s) \equiv 0.25$



Value function v_π

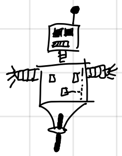
0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

Let's help the robot !

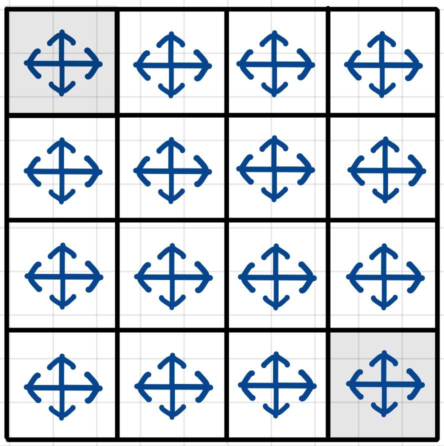
- the value function tells us "how good" a state s is under current policy π
- assume any state s
- assume you could "hack" the agent's policy for <u>one time step</u>, i.e. you can select the agent's <u>next action</u>
- after that, the agent resumes its policy
- which action do you pick?

How to Improve Policies?

Policy:



$\pi(a|s) \equiv 0.25$

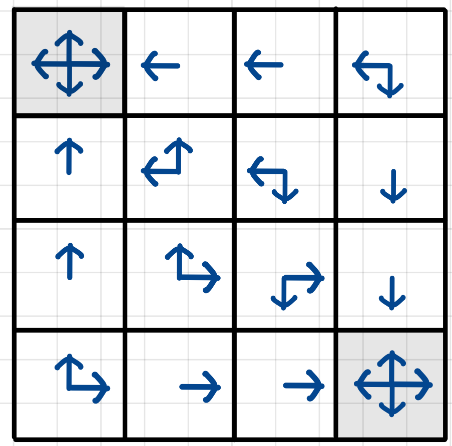


Value function V^{π}

0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

better: larger value

- we should move the agent to better states

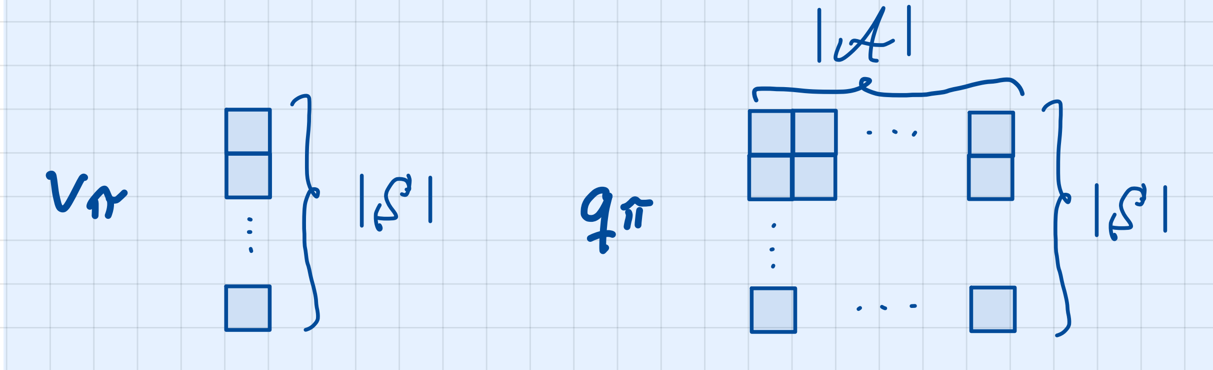


(multiple arrows denote equally good actions)

- we actually just came up with a new policy! (greedy policy)

- is it better than the old one?

The q Function



- to formalize greedy policies we introduce the q-function
- we have previously defined the value function

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

the q-function is the value function where we first pick an action of our choice:

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

also called **state-action value function**.

How to Compute q?

Method 1 v and q are related via

$$\boxed{v_\pi(s) = \sum_a \pi(a|s)\, q_\pi(s,a)} \qquad (1)$$

$$\boxed{q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a)\, v_\pi(s')} \qquad (2)$$

Thus, (2) allows to compute q_π from v_π

Method 2 Use Bellman equations in q_π

$$\boxed{q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \sum_{a'} \pi(a'|s')\, q_\pi(s',a')}$$

- Solve system of $|S| \times |A|$ equations, or
- do iterative Bellman updates

Greedy Policy

- let π be any policy and q_π its q-function
- define a new <u>deterministic</u> policy π':

$$\pi'(a \mid s) := \begin{cases} 1 & \text{if } a = \arg\max_{a'} q_\pi(s, a') \\ 0 & \text{otherwise} \end{cases}$$

(pick arbitrary $a' \in \arg\max$ if maximizer is not unique)

$\pi'(a \mid s)$ is called the <u>**greedy policy**</u> w.r.t. q_π (v_π)

Example: Grid World

<u>Value function</u>

0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

<u>Greedy Policy</u>

←↑→↓	←	←	←↓
↑	←↑	←↓	↓
↑	↑→	↓→	↓
↑→	→	→	←↑→↓

(multiple arrows : non-unique argmax)

is it better than the old one?

Policy Improvement Theorem

Let π be any policy and π' the (a) greedy policy w.r.t. v_π (q_π).
Then

$$v_\pi(s) \le v_{\pi'}(s) \qquad \forall s \in \mathcal{S}$$

Recall: $\boxed{v_\pi(s) = \sum_a \pi(a|s)\, q_\pi(s,a)}$ $\boxed{q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a)\, v_\pi(s')}$

$$\boxed{v_\pi(s)} = \sum_a \pi(a|s)\, q_\pi(s,a)$$

$$\le \underline{\max_a}\; q_\pi(s,a) \qquad \text{// use } \underline{\pi'} \text{ once, then } \pi$$

$$= \max_a\; r(s,a) + \gamma \sum_{s'} p(s'|s,a) \boxed{v_\pi(s')} \qquad \text{// repeat argument}$$

$$\le \underline{\max_a}\; r(s,a) + \gamma \sum_{s'} p(s'|s,a) \underline{\max_{a'}}\; q_\pi(s',a') \qquad \text{// use } \underline{\pi'} \text{ twice, then } \pi$$

$$= \max_a\; r(s,a) + \gamma \sum_{s'} p(s'|s,a) \left(\max_{a'}\; r(s',a') + \gamma \sum_{s''} p(s''|s',a') \boxed{v_\pi(s'')} \right)$$

$$\le \;\dots\; \le \boxed{v_{\pi'}(s)} \qquad \text{// by induction, } \pi' \text{ is at least as good as } \pi \qquad \square$$

Policy Iteration

We might apply the greedy policy iteratively, leading to policy iteration

- intialize $\hat{\pi}_0$, $k \leftarrow 0$
- repeat

  (1) compute $q_{\hat{\pi}_k}$ // e.g. with iterative policy evaluation

  (2) let $\hat{\pi}_{k+1}(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a'} q_{\hat{\pi}_k}(a', s) \\ 0 & \text{otherwise} \end{cases}$ // greedy policy

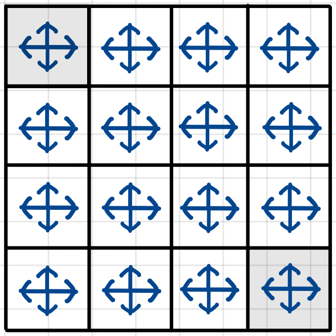
  - if $q_{\hat{\pi}_{k+1}} \approx q_{\hat{\pi}_k} \longrightarrow$ break
  - $k \leftarrow k+1$

Example: Grid World

$\hat{\pi}_k$

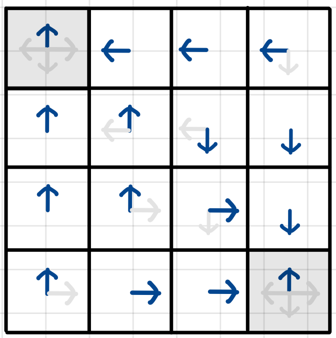
$v_{\hat{\pi}_k}$ $\left(q_{\hat{\pi}_k} = r(s,a) + \gamma \, \mathbb{E}_{s_{t+1}}[v_{\hat{\pi}_k}(s_{t+1})] \right)$

$k=0$



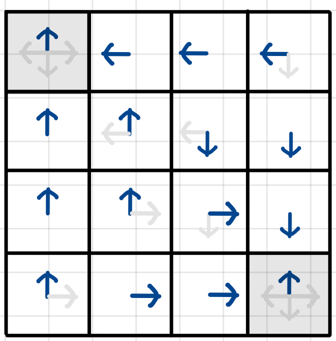
0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

$k=1$

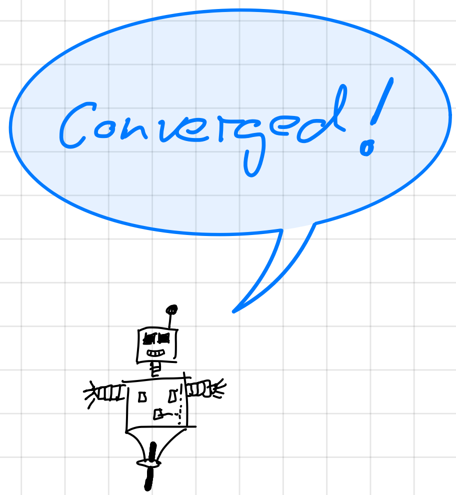


0	-1	-2	-3
-1	-2	-3	-2
-2	-3	-2	-1
-3	-2	-1	0

$k=2$



0	-1	-2	-3
-1	-2	-3	-2
-2	-3	-2	-1
-3	-2	-1	0



This policy seems globally optimal!

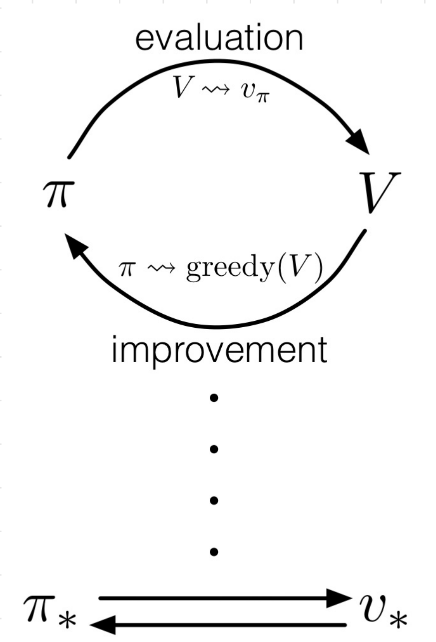
Does this always happen?

Optimal Policies

A policy π^* is optimal if $v_{\pi^*}(s) \geq v_\pi(s)$, $\forall \pi, s \in \mathcal{S}$

Theorem: Policy Iteration Converges to Optimal Policy

For any initial policy π_0 policy iteration converges to an optimal policy



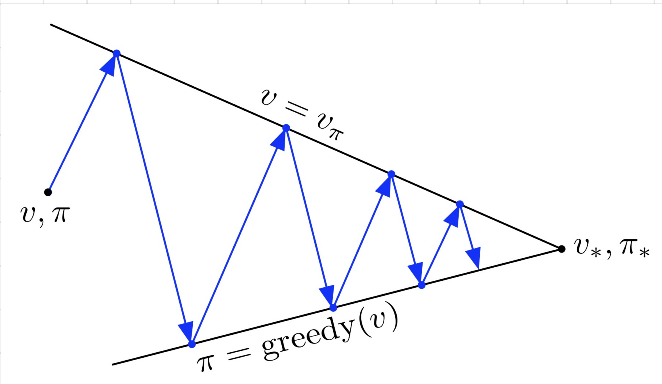
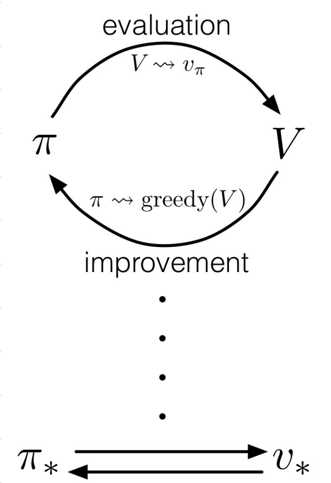
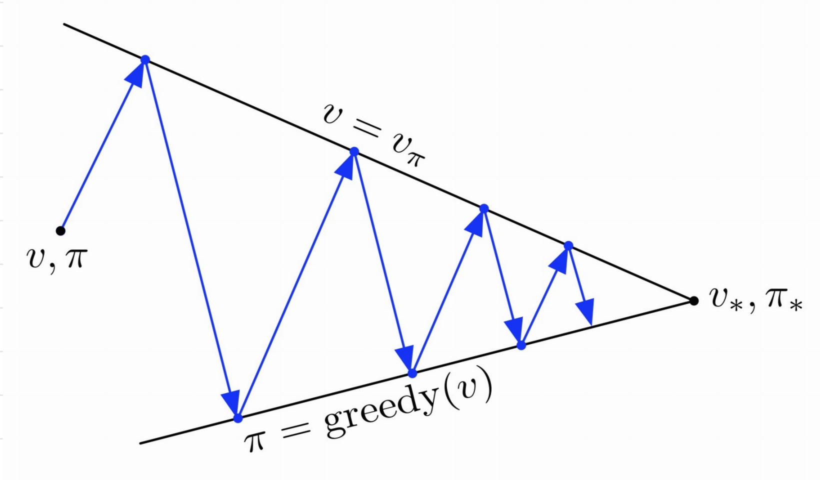


Image: Sutton & Barto

Efficiency of Policy Iteration?

Image: Sutton & Barto





- policy evaluation is expensive (in particular closed form)
- Iterative Policy Evaluation only converges for $k \to \infty$
- before that, intermediate results are typically not even a value function of any policy !
- how many iterations of Iterative Policy Evaluation do we really need?

Grid World

$\pi(a|s) = 0.25$

Iterative Policy Evaluation

$K=0$

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

$K=1$

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

$K=2$

0	-1.7	-2	-2
-1.7	-2	-2	-2
-2	-2	-2	-1.7
-2	-2	-1.7	0

$K=3$

0	-2.4	-2.9	-3
-2.4	-2.8	-3	-2.9
-2.9	-3	-2.8	-2.4
-3	-2.9	-2.4	0

...

$K=10$

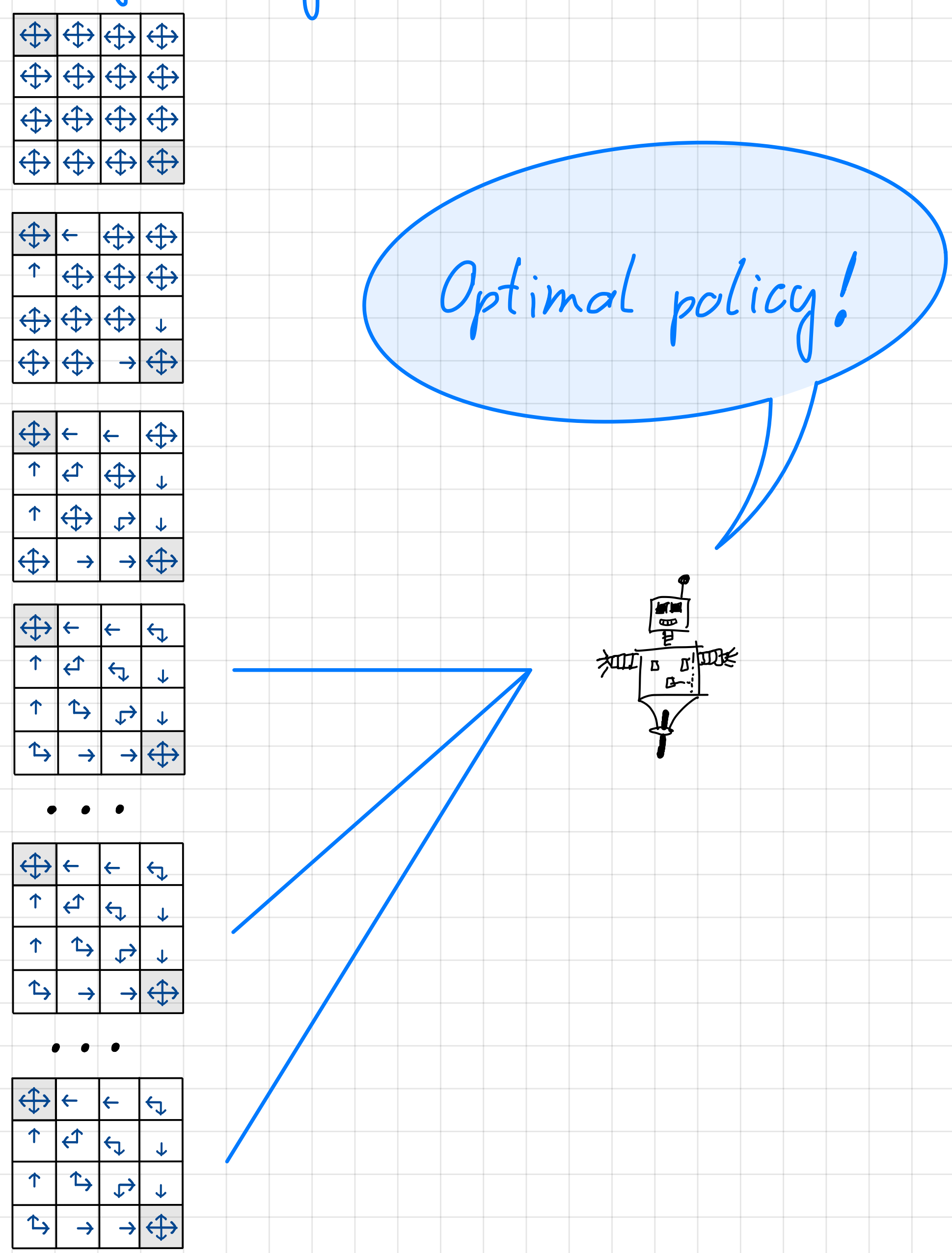
0	-6.1	-8.3	-8.9
-6.1	-7.7	-8.4	-8.3
-8.3	-8.4	-7.7	-6.1
-8.9	-8.3	-6.1	0

...

$K=\infty$

0	-13.8	-19.6	-21.6
-13.8	-17.7	-19.6	-19.6
-19.6	-19.6	-17.7	-13.8
-21.6	-19.6	-13.8	0

Greedy Policy



Policy Iteration with Truncated Policy Evaluation

initialize π, v
repeat until convergence
  repeat for $\boxed{K \text{ steps}}$
    $\forall s: \quad v(s) \leftarrow r(s) + \gamma \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)\, v(s')$
  $\pi \leftarrow \text{greedy}(v)$

- this algorithm also converges, for any K, to an optimal policy — no need for exact policy evaluation
- which K will work best, in order to minimize total compute? Unknown, open problem ...
- special case $K = 1$ is called value iteration