Finite
Markov Decision Processes

Jeanine Wippermann

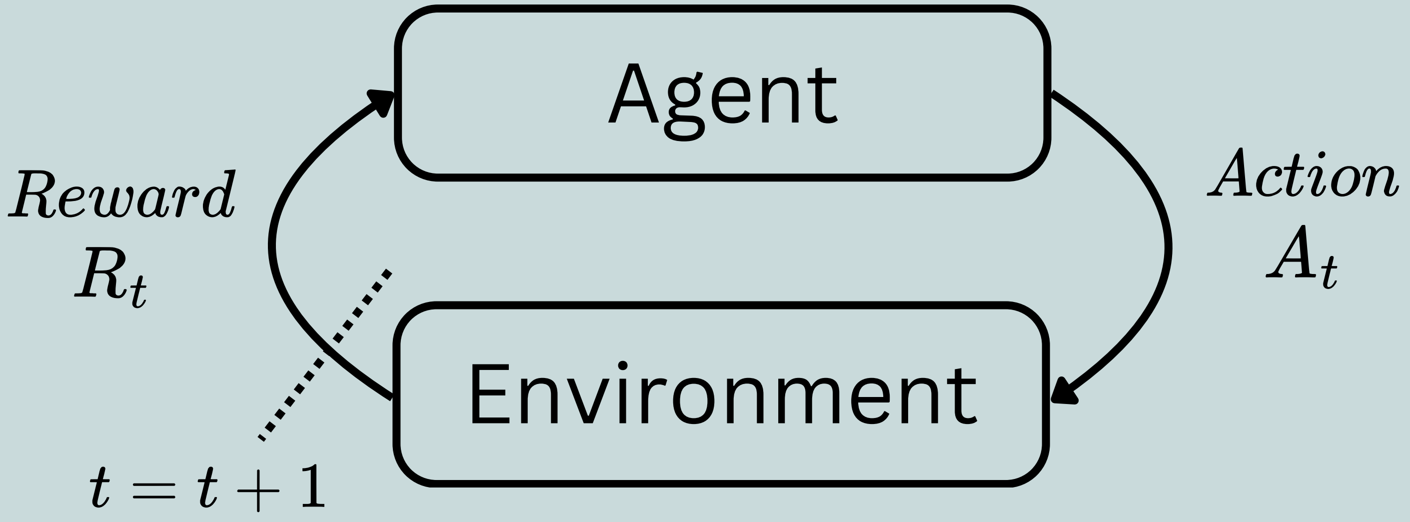
Content

- **Motivation**
- Defining a finite MDP
- Solving the finite MDP
- Finite MDP in real lifes

Motivation

define a suitable mathematical model for decision making processes

Reminder: k-armed Bandit



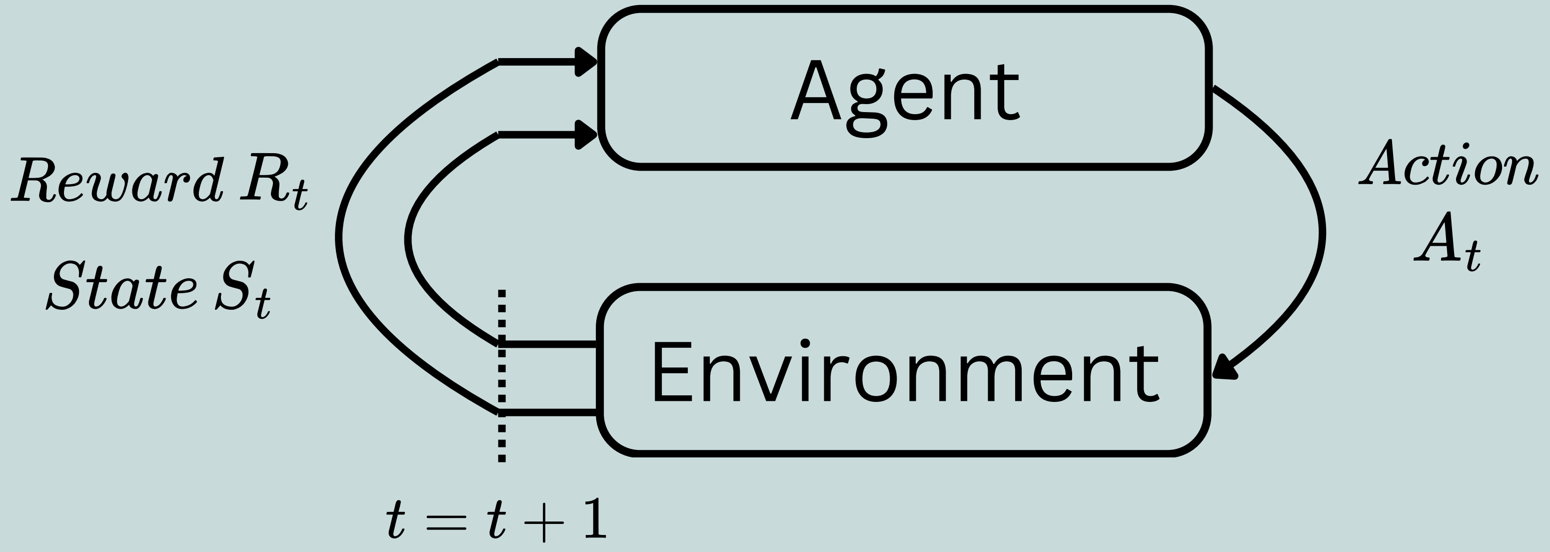
New: Markov Model

$$+\ State\ S_t$$

Content

- Motivation
- **Defining a finite MDP**
    - Markov decision process
    - Formal definition
    - Agent vs. Environment
    - Actions, states and rewards
    - Policy of the system
    - Dynamics of the system
- Solving the finite MDP
- Finite MDP in real lifes

Markov Decision Process



1. Agent chooses an action
2. Environment responds with a reward and transforms into a new state
3. Agent chooses again...

States need to have the **Markov Property:**

The next state is **independ of past states** and does only **depend on the current state and the action taken**

Example: Chess

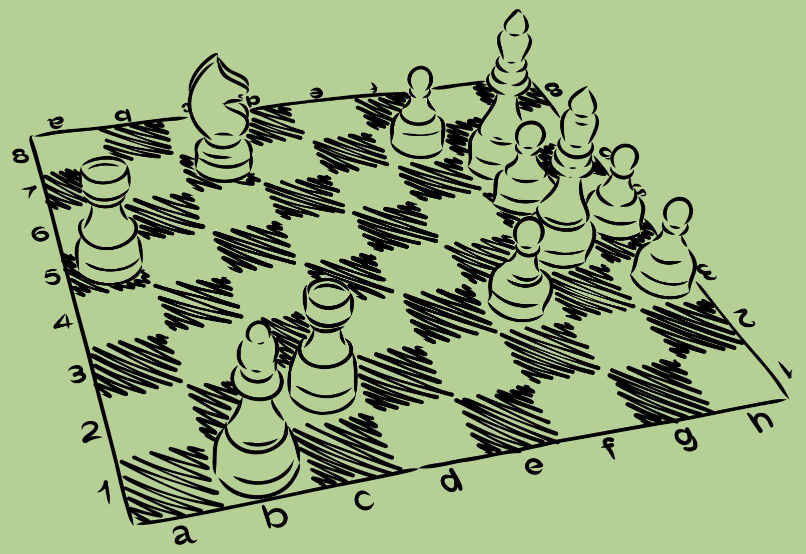
1. Agent decides for an action:
    - moving a figure

2. Environment responds with a reward and transforms into a new state:
    - chess position of the agent the opponent changes
    - agent has won, lost or the game continues

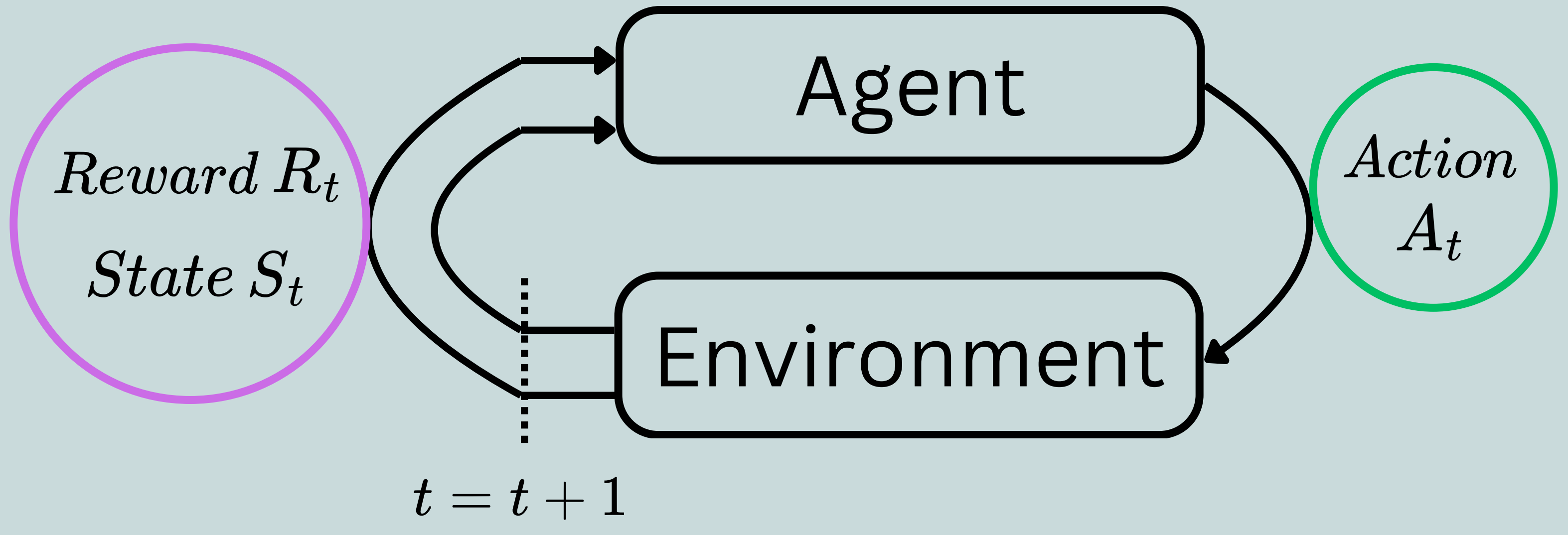
3. Agent makes a move again

Q: Do chess positions have the Markov property?

?



Formal definition



- A set of actions $\mathcal{A}$
- A set of states $\mathcal{S}$
- A set of rewards $\mathcal{R}$

contain a **finite** amount of elements

- The probabilistic **dynamic function** $p(\cdot|s_t, a_t)$

Reinforcement Learning task

Find the **optimal policy function** $\pi^*(a|s)$,

under which we get **maximal rewards**

Agent vs. Environment

- Decision maker
- does not need physical boundaries
- does have some information about the environment
- smallest possible unit

- Everything outside of the agent
- anything that cannot be charged arbitrarily by the agent

Actions, states & rewards

- any decision we want to learn how to make
- represented as an array
- made by the agent

- anything that might be useful for the decision making
- represented as an array
- belongs to the environment

- information *what* to achieve
- always numerical values
- belongs to the environment

Policy of the system

- is unknown when defining a Markov decision process

Given the **current state**:
predict the **probability of the chosen action**

$$\pi(a|s) \doteq Pr\{A_t = a | S_t = s\}$$

with the property: $\sum_{a \in \mathcal{A}(s)} \pi(a|s) = 1$ for all $s \in \mathcal{S}$

Dynamics of the system

- Needs to be modeled before defining a Markov decision process

Given the **current state-action pair**:
predict the **probability of the next state & reward**

$$\underbrace{p(s', r}_{}|\underbrace{s, a)}_{} \doteq Pr\{\underbrace{S_t = s', R_t = r}_{}|\underbrace{S_{t-1} = s, A_{t-1} = a}_{}\}$$

with the property: $\sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a) = 1$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$

State transition probability

Probability of the **next state**

$$p(s'|s,a) = \sum_{r\in\mathcal{R}} p(s',r|s,a)$$

Expected reward

for a **state-action** pair

$$r(s,a) = \sum_{r\in\mathcal{R}} r\, p(r|s,a)$$

Reward probability

Probability of the **next reward**

$$p(r|s,a) = \sum_{s'\in\mathcal{S}} p(s',r|s,a)$$

Expected reward

for a **state-action-next state** triple

$$r(s,a,s') = \sum_{r\in\mathcal{R}} r \frac{p(s',r|s,a)}{p(s'|s,a)}$$

Dynamics: $p(s',r|s,a) \doteq Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$

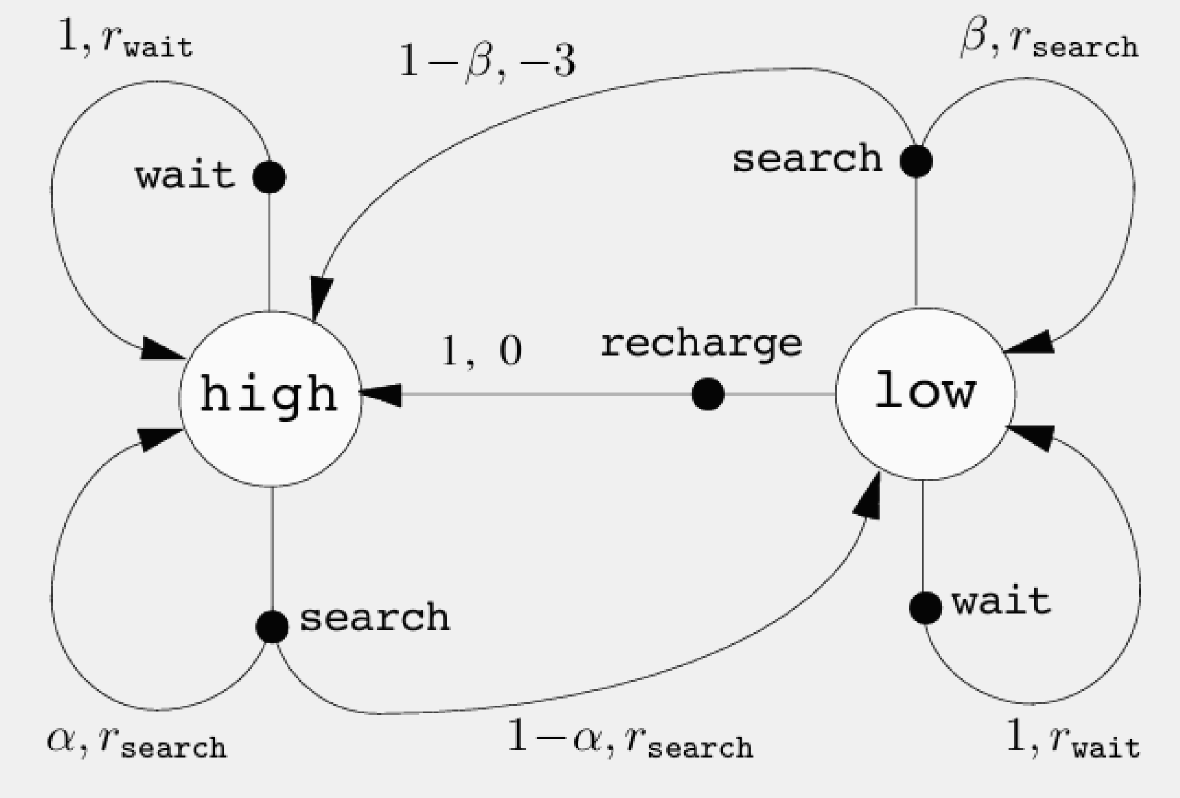
Example: Recycling robot

- **States:** low battery; high battery
- **Actions:** Search for a can (and loose battery); Wait for someone to bring a can;
  . go back and recharge battery
- **Reward:** positive if can is collected; negative if battery runs out; 0 else

Dynamics

s	a	s'	$p(s' \mid s, a)$	$r(s, a, s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1-\alpha$	$r_{\texttt{search}}$
low	search	high	$1-\beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	-
low	wait	high	0	-
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	-

Transition graph



Content

- Motivation
- Defining a finite MDP
- **Solving the finite MDP**
  - **Reward, return and value**
    - Reward hypothesis
    - Definitions
    - Episodic vs. continuing tasks
    - Discounted return
    - Value function
  - Bellman equation
  - Optimality
- Finite MDP in real lifes

Reward hypothesis

"All we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward)."

Definitions

Numerical value for immediate feedback

Reward r_t

Measurement for the amount of future rewards

Return $G_t = \sum_{t'>t} R_{t'}$

Expected reward, how good is our current state or action taken in the long term?

Value $\mathbb{E}[G_t]$

Task: Find the optimal policy function $\pi^*(a|s)$, under which we get **maximal rewards**

Episodic vs continuing task

Agent-environment-interaction break down into finite episodes

$$t_1, t_2, t_3, \ldots, t_T$$

$$G_t = \sum_{t'=t+1}^{T} R_{t'}$$

Agent-environment-interaction never ends

$$t_1, t_2, t_3, \ldots \to \infty$$

$$G_t = \sum_{t'=t+1}^{\infty} R_{t'} \stackrel{?}{=} \pm\infty$$

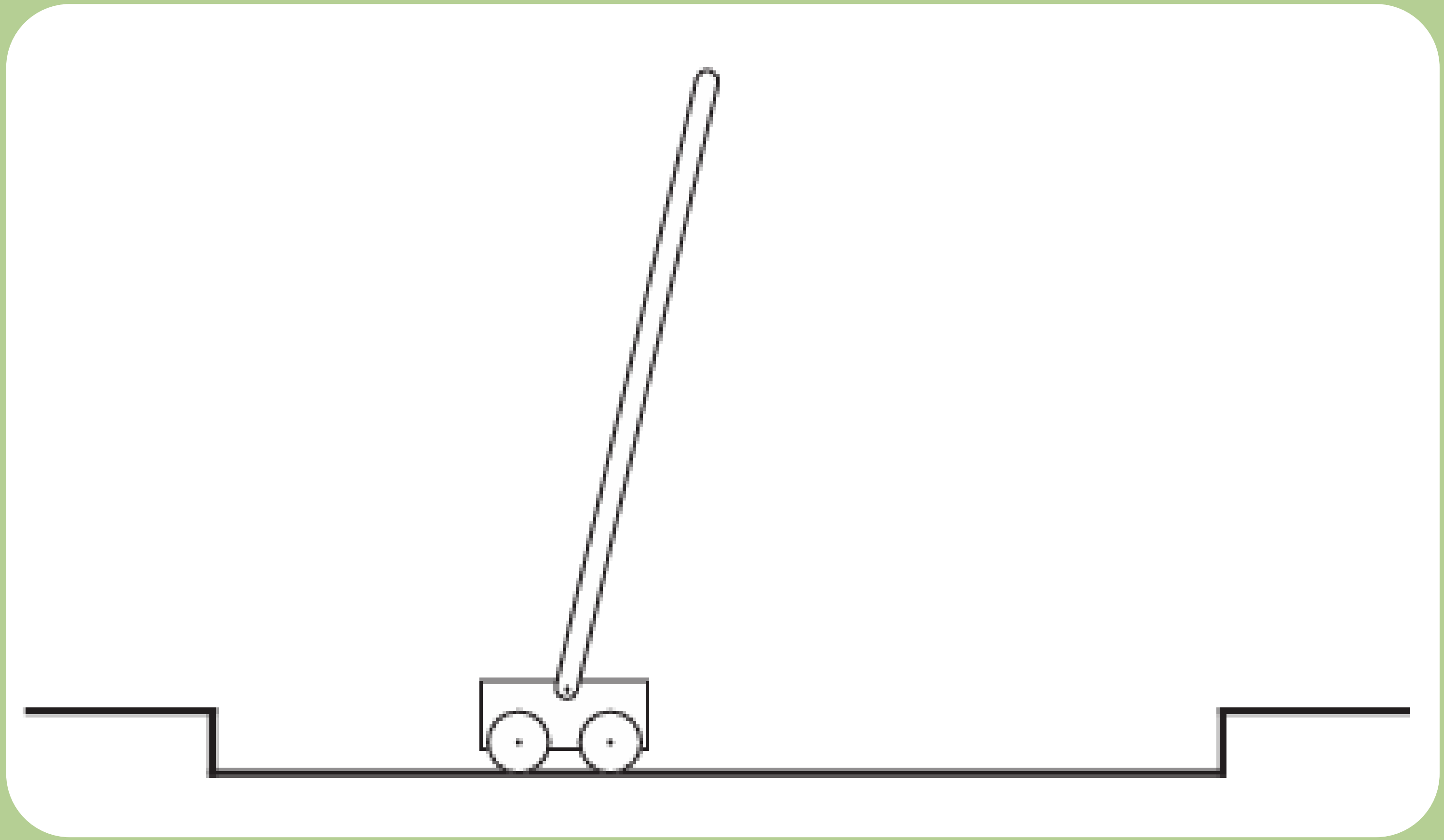
Example: Pole balancing

Goal: don't let the stick fall down!

If that happens: Start again with pole reset to vertical

Q: Is this task episodic or continues?

?



Example: Pole balancing

Goal: don't let the stick fall down!

If that happens: Start again with pole reset to vertical

Possibility 1:

Treated as an epsodic task:

When balancing fails, return is set to 0

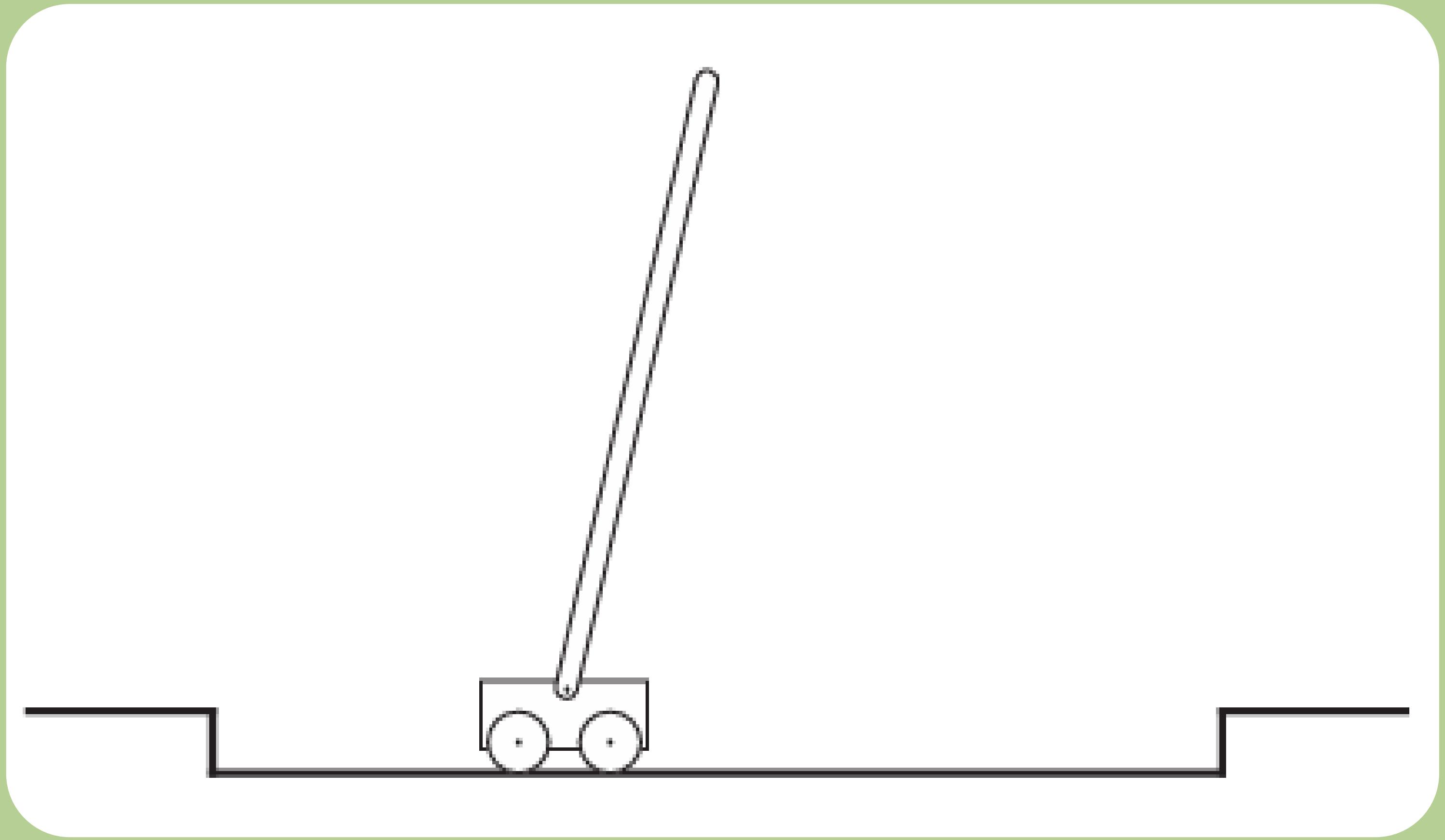
and task starts again

Possibility 2:

Treated as an continues task:

Return includes the reward of *all* future

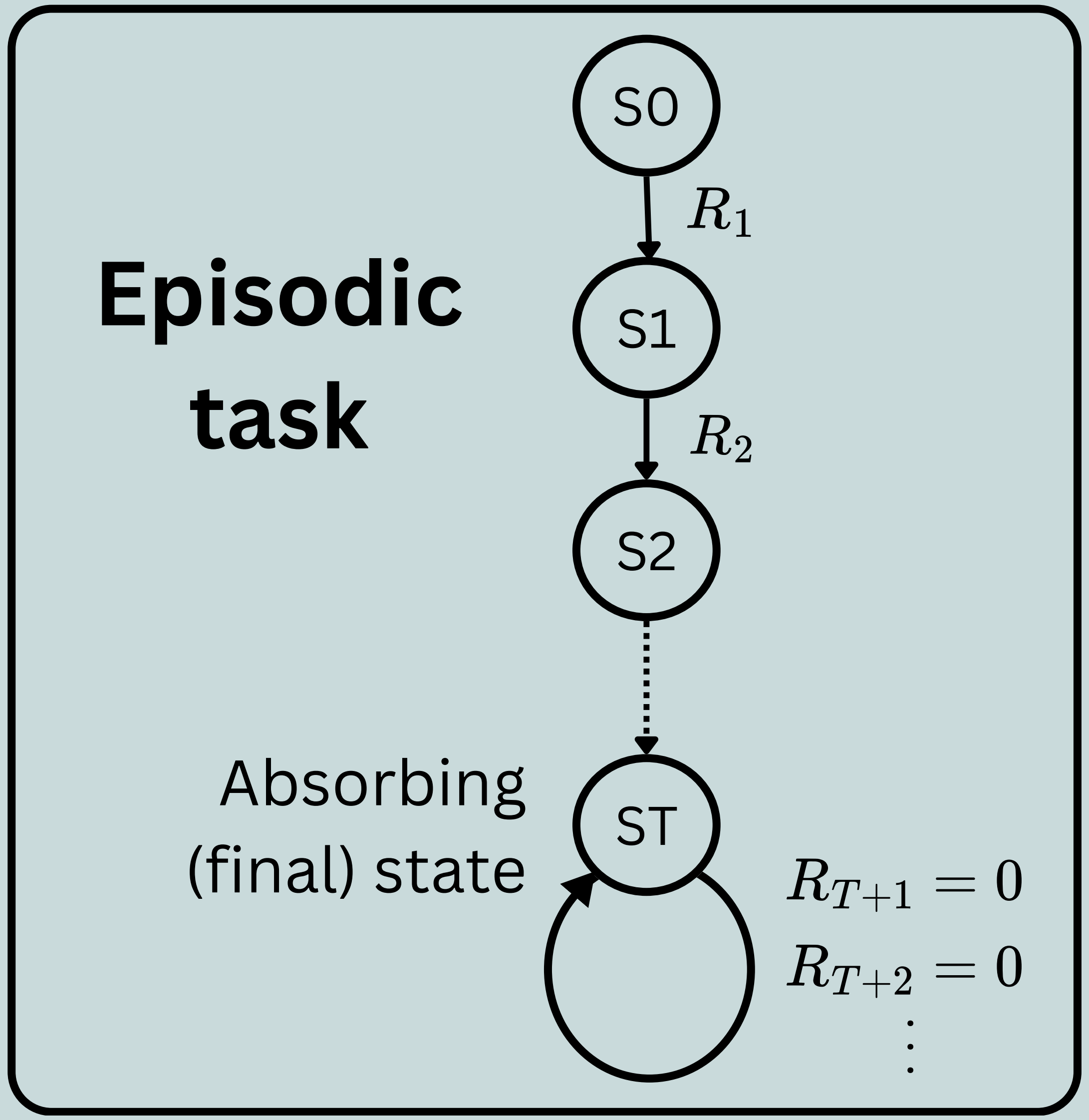
trials, failure gets a negative reward



Discounted return

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

- Discount rate γ, with $0 \leq \gamma \leq 1$
- Finite for all $\gamma < 1$



Return: Measurement for the amount of future rewards

Example: Continues pole balancing

Reward: **+1** for balancing, **0** for failure

"Classic" return

$$G_t = \sum_{t' > t} R_{t'}$$

$$= \sum_{t' > t \wedge R_t' = 1}^{\infty} 1 + \sum_{t' > t \wedge R_t' = 0}^{\infty} 0$$

$$= \infty$$

Discounted return

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

$$\leq \sum_{k=0}^{\infty} \gamma^k$$

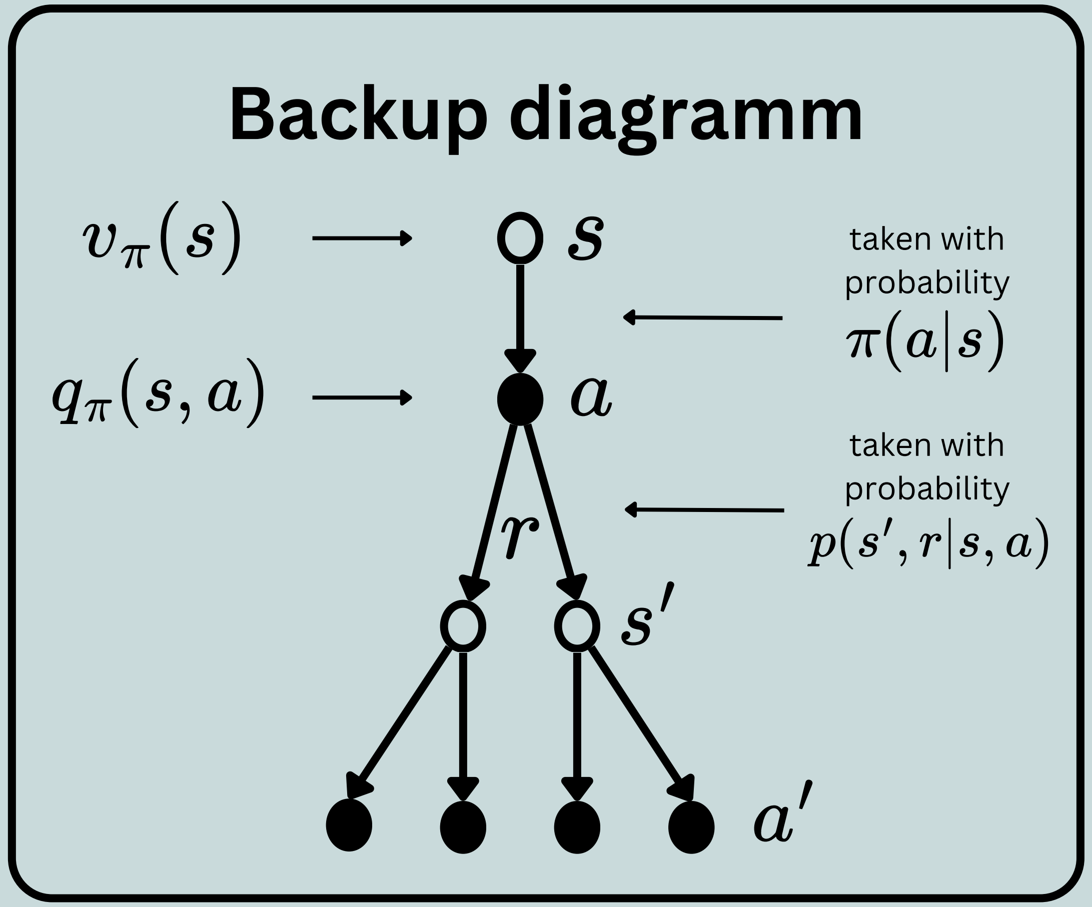
$$= \frac{1}{1-\gamma}$$

Value function

State-value function for policy π

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

Expected Return when **starting in state s** and following policy π



Action-value function for policy π

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

Expected Return when **starting in state s, choosing action a** and following policy π

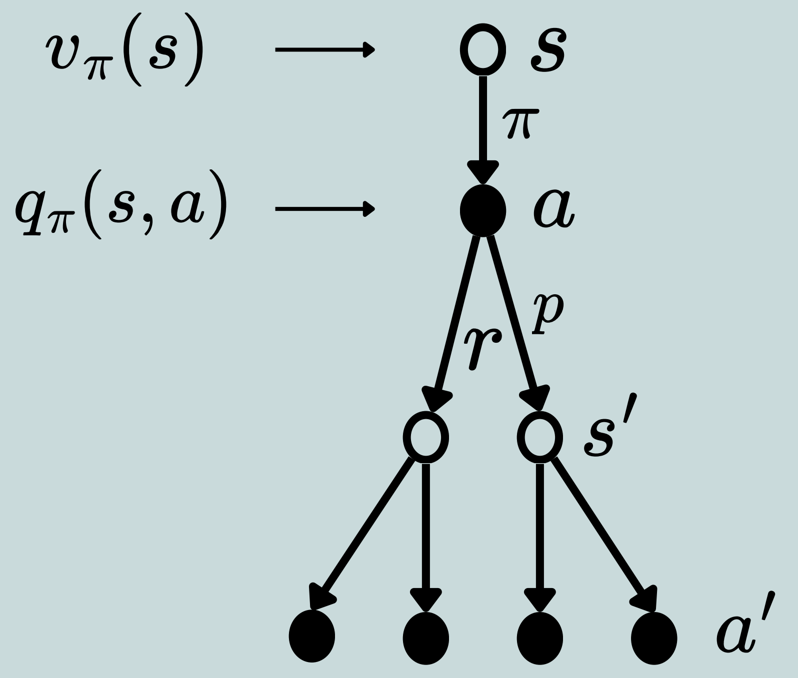
$$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \qquad v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = R_{t+1} + \gamma G_{t+1}$$

Connection between state-value and action-value

$$v_\pi(s) = \sum_{a \in \mathcal{A}(s)} \pi(a|s) q_\pi(s,a)$$

Backup diagramm



Recursive property

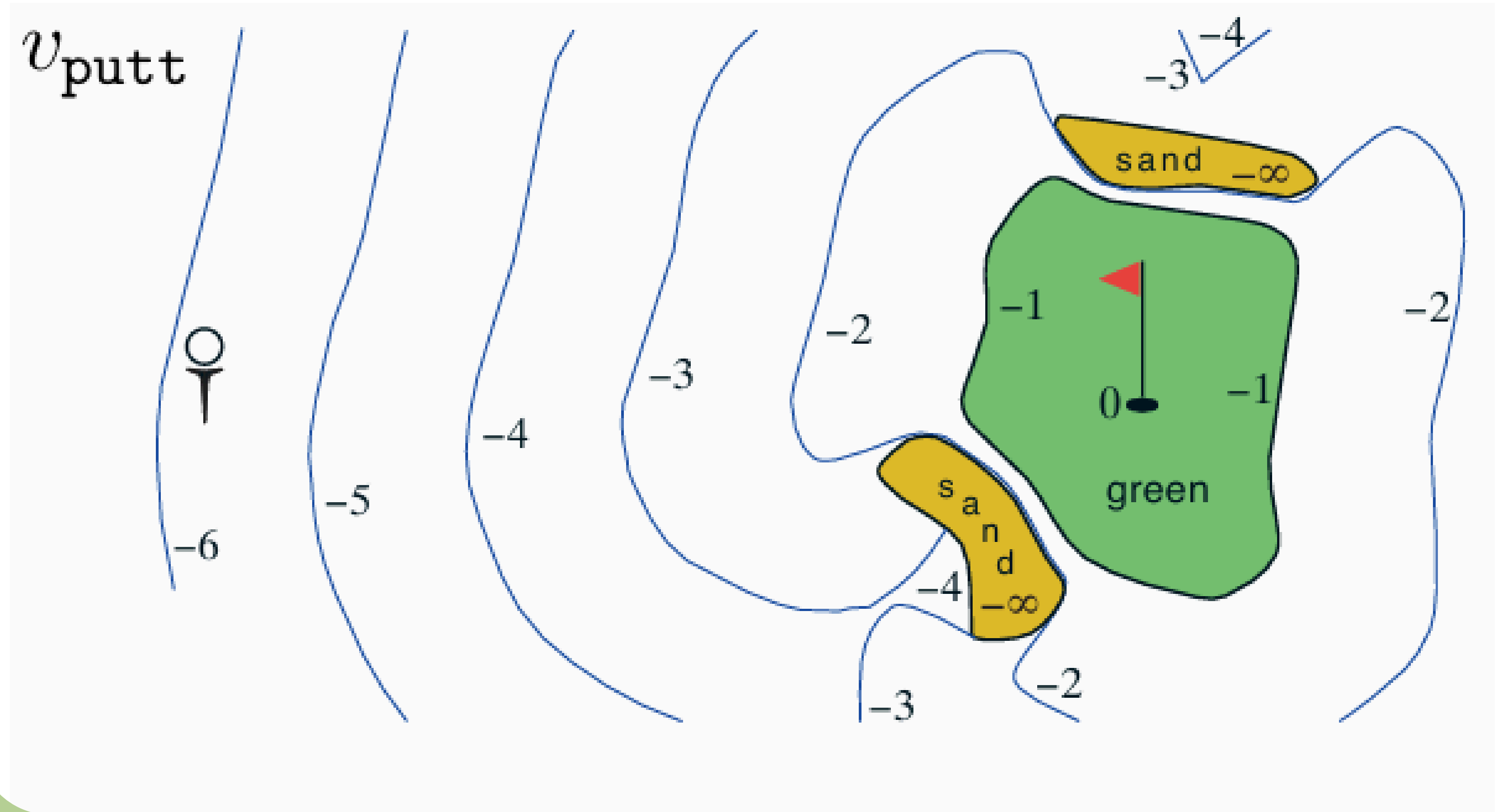
$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s]$$

$$q_\pi(s,a) = \mathbb{E}_\pi[r + \gamma v_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

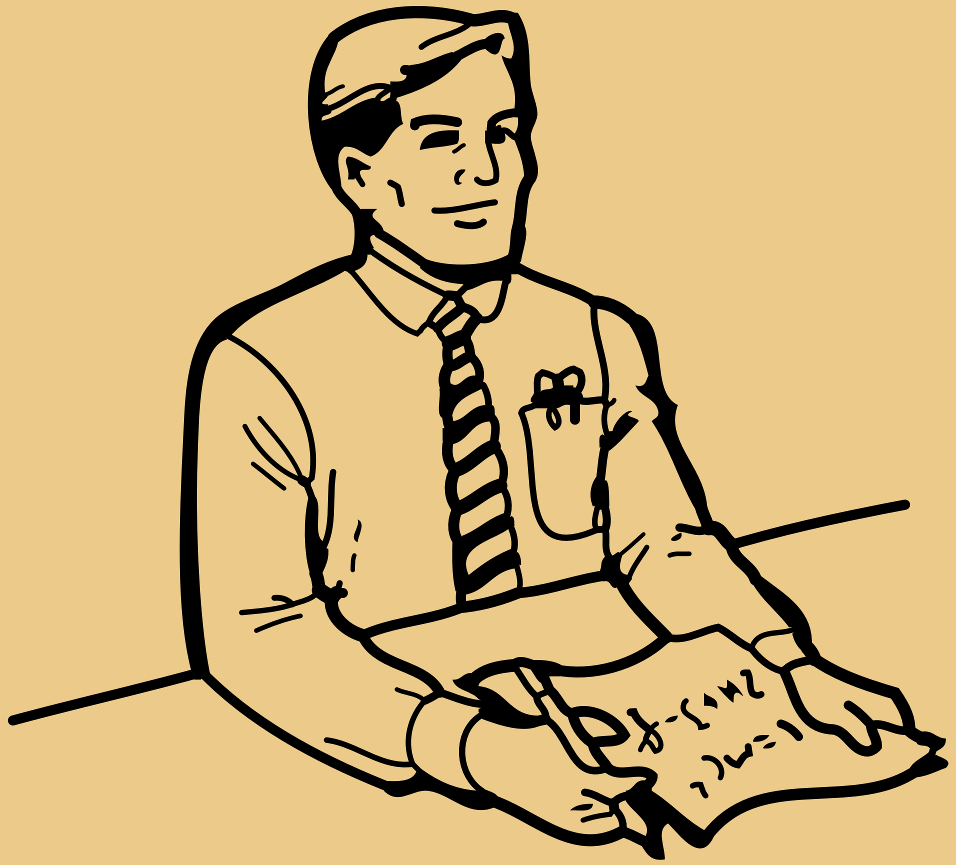
Example: golf

- **States:** location
- **Actions:** how we aim and swing at the ball, which club we select (putter or driver)
- **Reward:** -1 each stroke until we hit the ball into the hole

State-Value function for the policy of always using the putter (and striking in the direction of the hole)



Content



- Motivation
- Defining a finite MDP

- **Solving the finite MDP**
  - Reward, return and value
  - **Bellman equation**
  - Optimality

- Finite MDP in real lifes

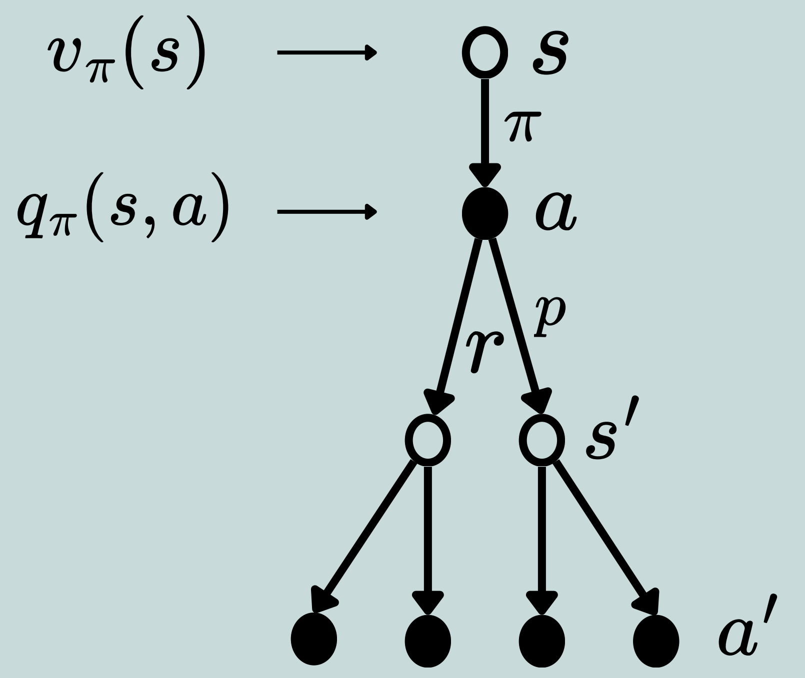
Bellman equation

- Goal: Write down the expected return expicitely

Q: How can we write down the expected return expicitely?

?

Backup diagramm



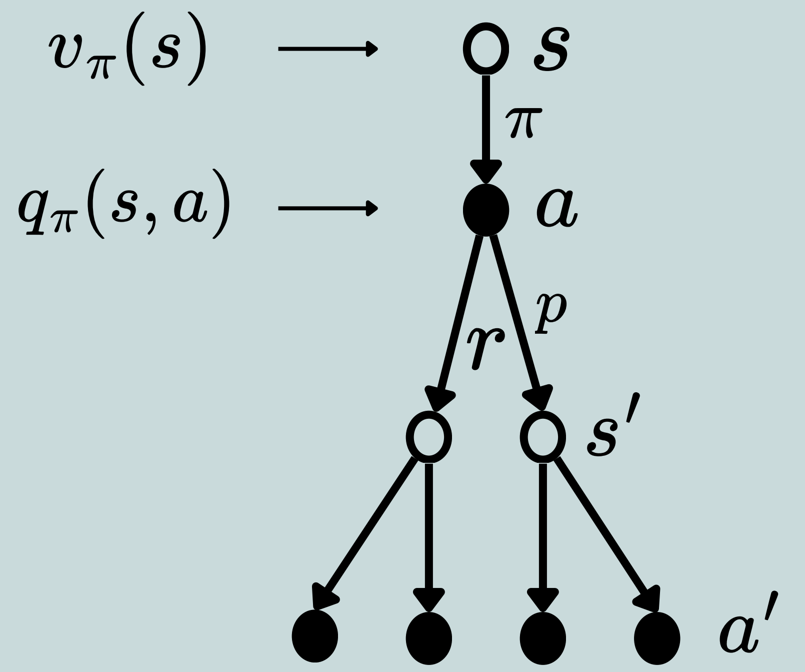
$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s] \qquad q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$$

Bellman equation

- Goal: Write down the expected return explicitely

1. take the possibility of one future path
2. multiply by its return
3. sum over all possible future paths

Backup diagramm



$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s] \qquad q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$$

Bellman equation

- use recursive behavior of the value functions

for the **state value**

$$v_\pi(s) = \sum_{a \in \mathcal{A}(s)} \pi(a|s) \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

Q: What does the Bellman equation of the action-value look like?

?

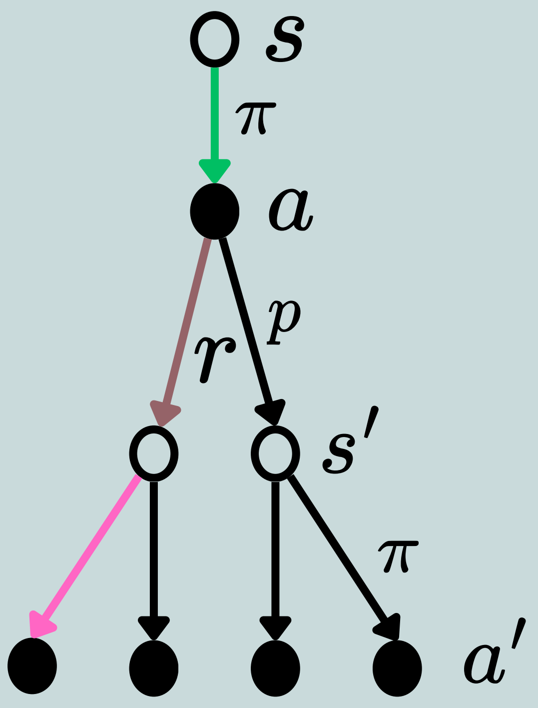
Backup diagramm

$v_\pi(s)$

$q_\pi(s, a)$

$v_\pi(s')$

$q_\pi(s', a')$



$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$$

Bellman equation

- use recursive behavior of the value functions

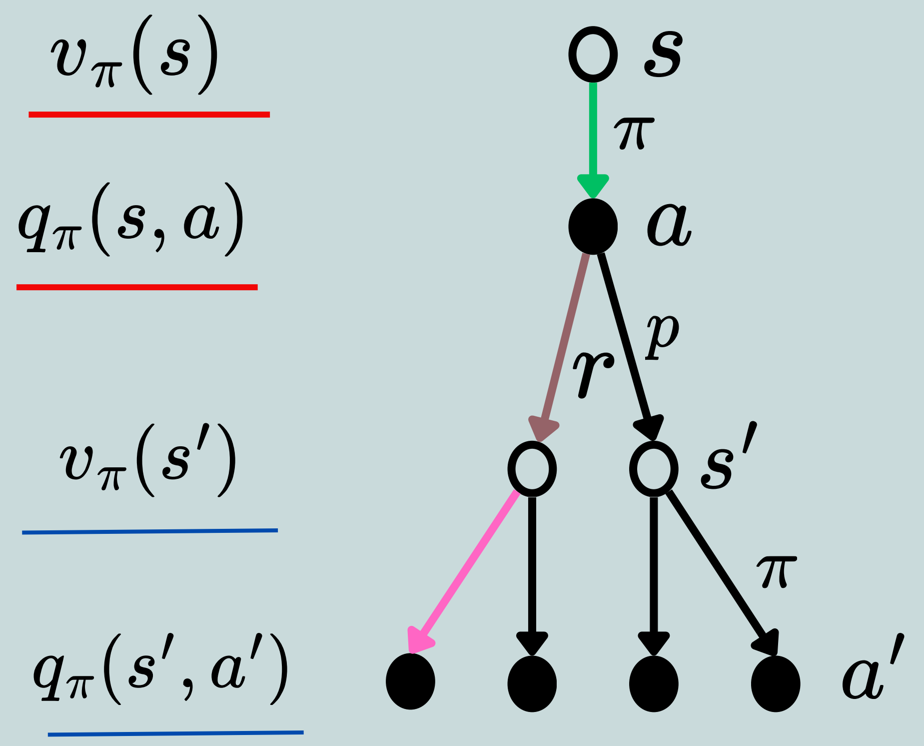
for the **state value**

$$v_\pi(s) = \sum_{a \in \mathcal{A}(s)} \pi(a|s) \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

for the **action-value**

$$q_\pi(s, a) = \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a) \sum_{a' \in \mathcal{A}(s')} \pi(a'|s')[r + \gamma q_\pi(s', a')]$$

Backup diagramm



Bellman equation

- Normally use the state-action Bellman equation

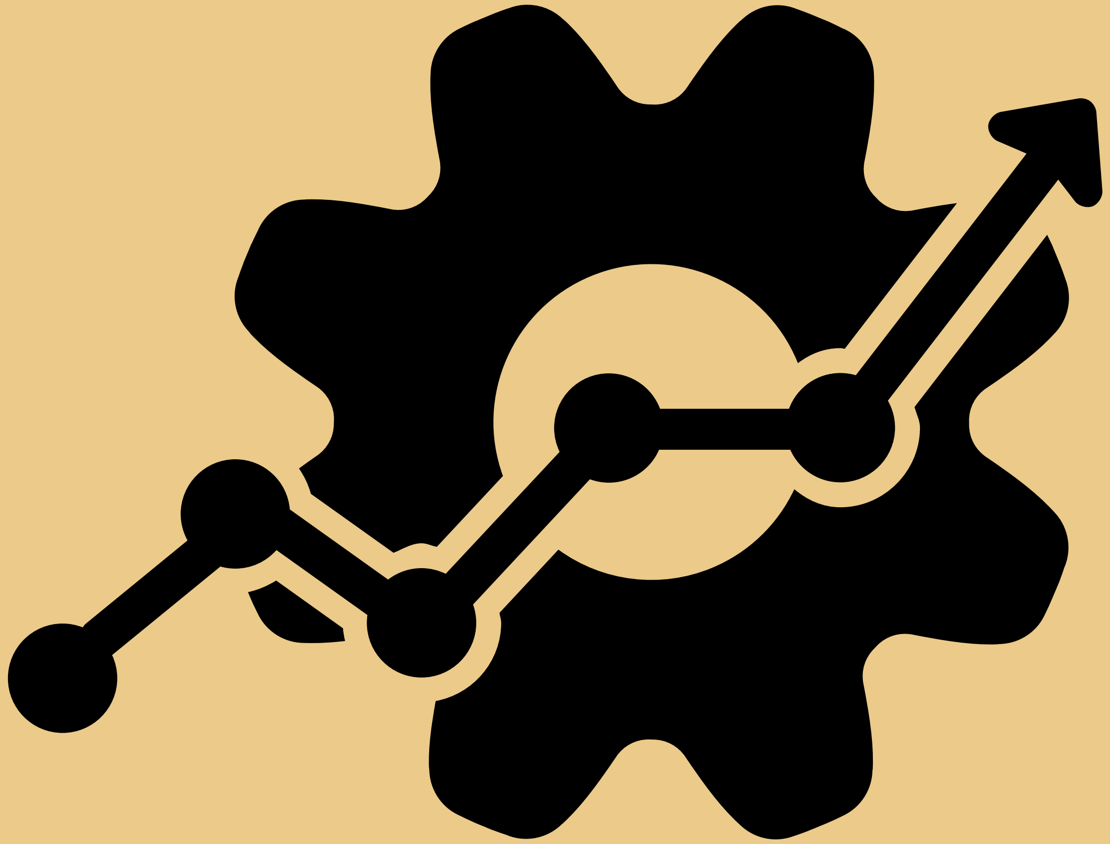
$$\underbrace{v_\pi(s)}_{\substack{\text{expected current return}\\ \text{(when starting in}\\ \text{state s)}}} = \sum_{a\in\mathcal{A}(s)} \underbrace{\pi(a|s)}_{\substack{\text{probability, that}\\ \text{we currently}\\ \text{choose action a}}} \sum_{s'\in\mathcal{S},r\in\mathcal{R}} \underbrace{p(s',r|s,a)}_{\substack{\text{probability, that}\\ \text{we get state s'}\\ \text{and reward r next}}} \underbrace{[r+\gamma \overbrace{v_\pi(s')}^{\substack{\text{Expected}\\ \text{next return}}}]}_{\substack{\text{expected current return}\\ \text{(when state s' and}\\ \text{reward r come next)}}}$$

Bellman equation

- excplicit recursive expression of the value function
- relationship between the value of the current state and the value of the following states
- **Now: find a policy, that maximizes the Bellman equation**

$$v_\pi(s) = \sum_{a \in \mathcal{A}(s)} \pi(a|s) \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

Content



- Motivation
- Defining a finite MDP
- **Solving the finite MDP**
  - Reward, return and value
  - Bellman equation
  - **Optimality**
    - Optimal state-value function
    - Optimal Bellman equation
    - Optimal policy
- Finite MDP in real lifes

Optimal value function

What is the *maximal* value of a given state/action we can expect?

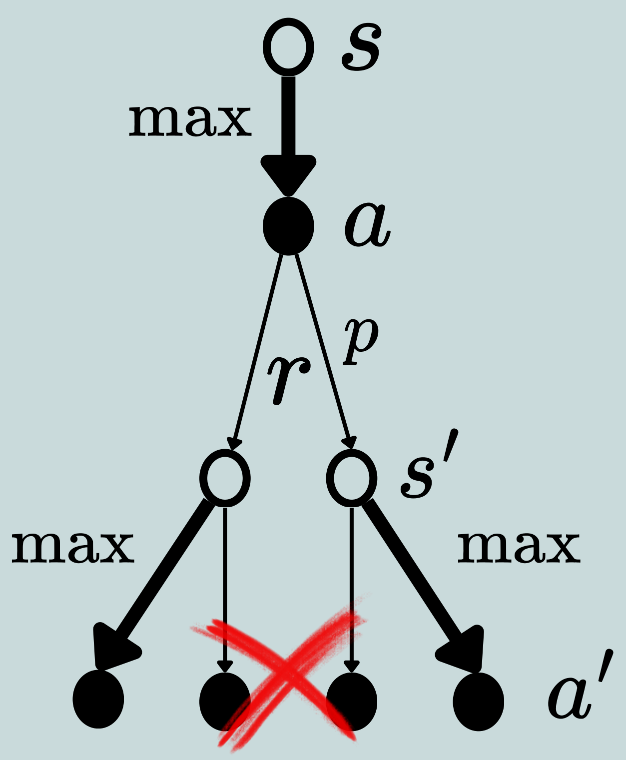
Optimal state-value function

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

Optimal action-value function

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

Backup diagramm for $v_*(s)$



choose action with maximal value

Optimal Bellmann equation

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

- System of nonlinear equations, one equation per state
- Solve for unknowns $v_*(s_1), v_*(s_2), \ldots, v_*(s_n)$
- all n unknowns in all n equations
- actions that maximize the right side are optimal actions
- optimal actions define our optimal policy $\pi_*(a|s)$

Bellmann equation $v_\pi(s) = \sum_{a \in \mathcal{A}(s)} \pi(a|s) \sum_{s' \in \mathcal{S}, r \in \mathcal{R}} p(s', r|s, a)[r + \gamma v_\pi(s')]$

Example: Optimal Bellmann equation

MDP: $\mathcal{S} = (S_1, S_2)$ $\mathcal{A} = (A_1, A_2)$ $\mathcal{R} = (R_1, R_2)$ $p(S_i, R_j | S_k, A_l) = C_{ijkl}$

$$v_*(S_1) = \max \begin{cases} (C_{1111} + C_{1211})\gamma v_*(S_1) + (C_{2111} + C_{2211})\gamma v_*(S_2) + (C_{1111} + C_{2111})R_1 + (C_{1211} + C_{2211})R_2 \\ (C_{1112} + C_{1212})\gamma v_*(S_1) + (C_{2112} + C_{2212})\gamma v_*(S_2) + (C_{1112} + C_{2112})R_1 + (C_{1212} + C_{2212})R_2 \end{cases}$$

$$v_*(S_2) = \max \begin{cases} (C_{1121} + C_{1221})\gamma v_*(S_1) + (C_{2121} + C_{2221})\gamma v_*(S_2) + (C_{1121} + C_{2121})R_1 + (C_{1221} + C_{2221})R_2 \\ (C_{1122} + C_{1222})\gamma v_*(S_1) + (C_{2122} + C_{2222})\gamma v_*(S_2) + (C_{1122} + C_{2122})R_1 + (C_{1222} + C_{2222})R_2 \end{cases}$$

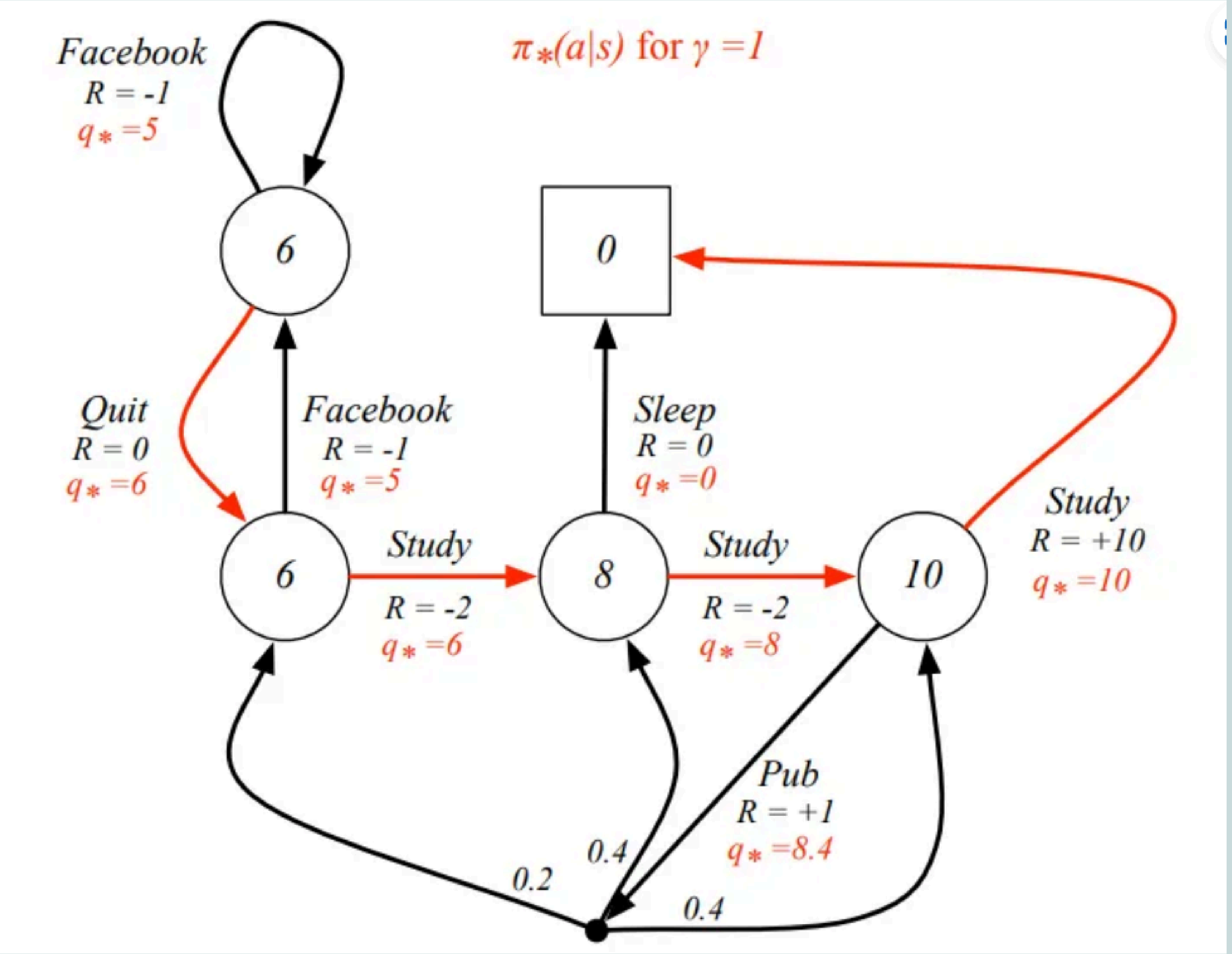
***Short*:** $v_*(S_1) = \max \begin{cases} d_1 v_*(S_1) + d_2 v_*(S_2) + d_3 \\ d_4 v_*(S_1) + d_5 v_*(S_2) + d_6 \end{cases}$ $v_*(S_2) = \max \begin{cases} d_7 v_*(S_1) + d_8 v_*(S_2) + d_9 \\ d_{10} v_*(S_1) + d_{11} v_*(S_2) + d_{12} \end{cases}$

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

Optimal policy

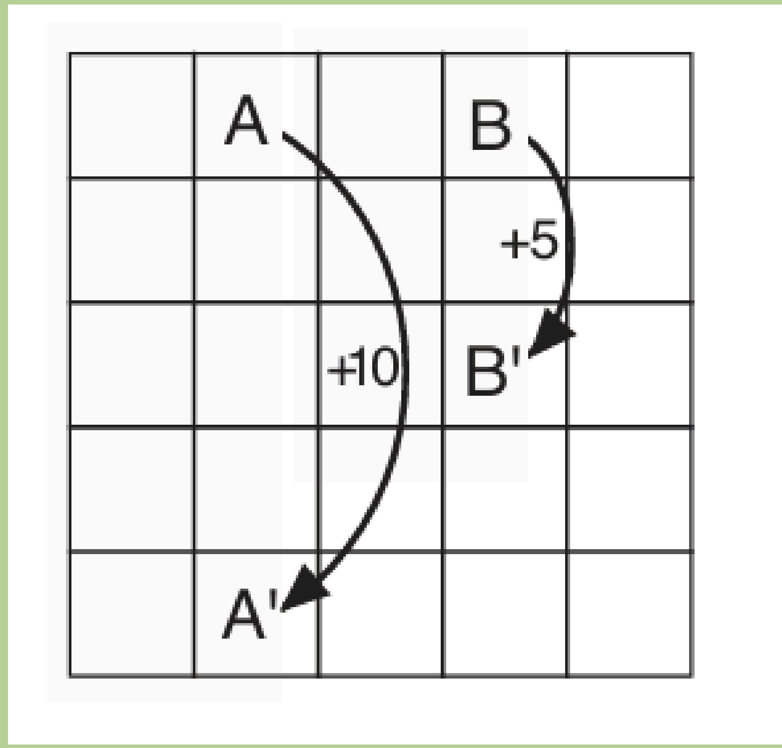
Simple rule:
always choose the action with
maximal possible value!

$$\pi_*(a|s) = \begin{cases} 1 \text{ if } a = \arg\max\limits_{a \in \mathcal{A}(s)} q_*(s,a) \\ 0 \text{ else} \end{cases}$$



Example: Gridworld

- **States:** position on the grid
- **Actions:** go up, down, right or left
- **Rewards:**
  - 10 on A -> agent is transfered to A’
  - 5 on B -> agent is transfered to B’
  - -1 when position is outside of the grid



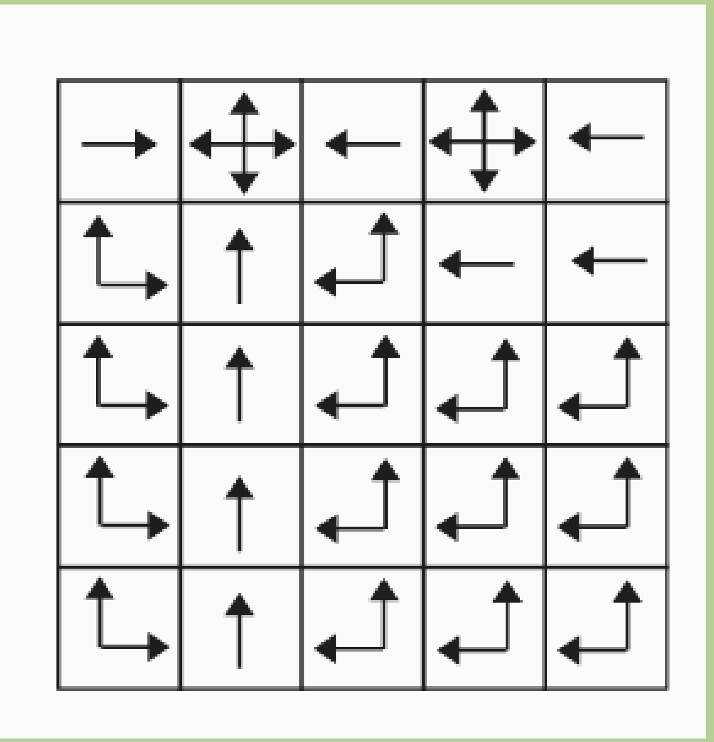
Gridworld

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

State-values for random policy

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

Optimal state-values



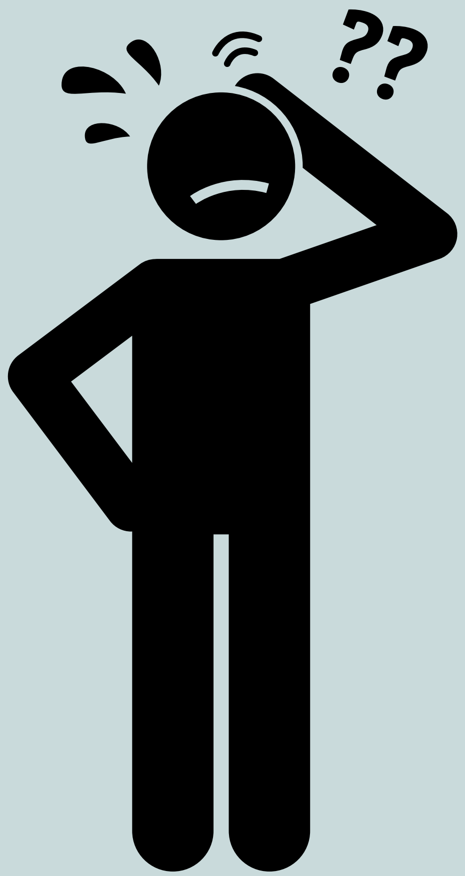
Optimal policy

Content

- Motivation
- Defining a finite MDP
- Solving the finite MDP

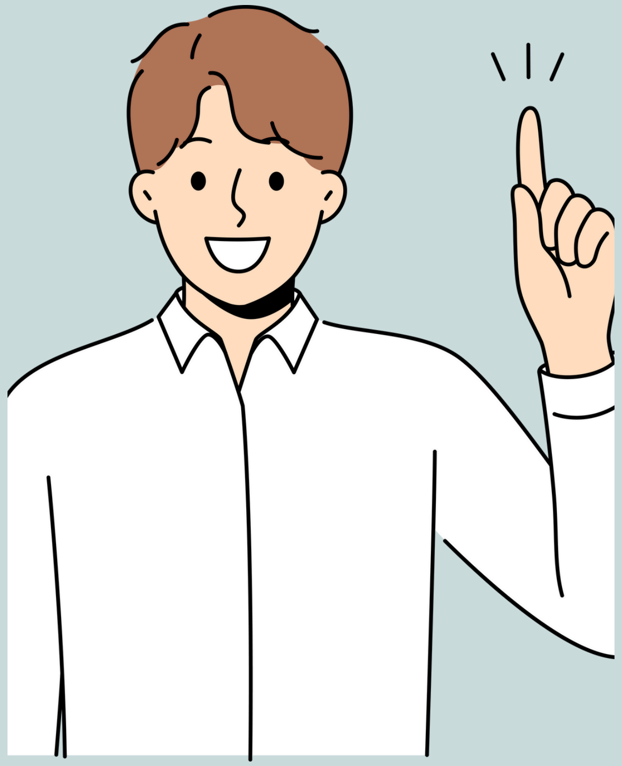
- **Finite MDPs in real life**
    - Problems in real life
    - Solutions in real life

Problems in real life



- dynamics of the environment are not completely known
- states do not have the Markov property
- computational resources are insufficient

Solutions in real life



- approximation, approximation, approximation... for example:
    - parameterized function representation
    - ignoring states with low probability

Thank you for your attention!

References: Reinforcement learning, an Introduction,
by Richard S. Sutton and Andrew G. Barto