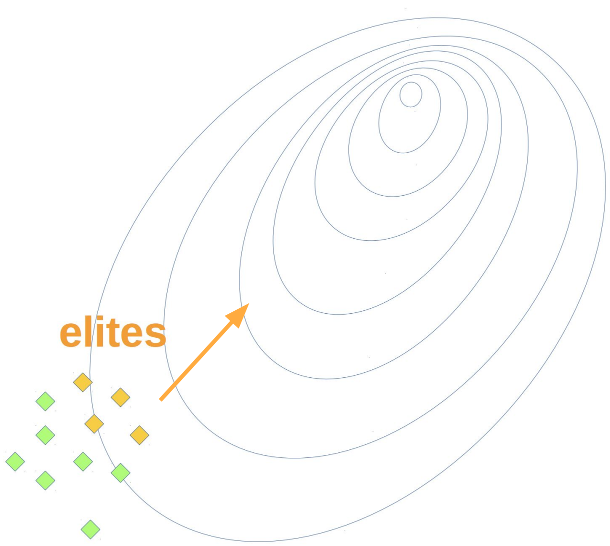
Practical RL – Week 2

Shvechikov Pavel

Previously in the course

- The MDP formalism
  - State, Action, Reward, next State
- Cross-Entropy Method (CEM)
  - easy to implement
  - competitive results
  - black box
    - no knowledge of environment
    - no knowledge of intermediate rewards



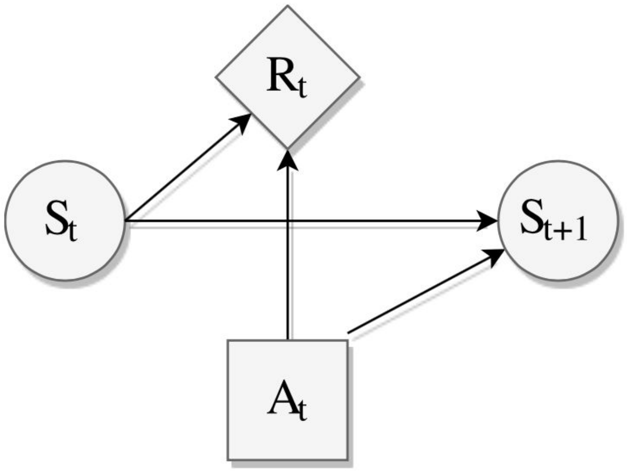
Improve on the CEM → dive into the black box

Provided we know all, how to find an optimal policy?

Definition of Markov Decision Process

MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where

1. $\mathcal{S}$ – set of states of the world
2. $\mathcal{A}$ – set of actions
3. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \triangle(\mathcal{S})$ – state-transition function, giving us $p(s_{t+1} \mid s_t, a_t)$
4. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ – reward function, giving us $\mathbb{E}_R \left[R(s_t, a_t) \mid s_t, a_t \right]$.



Markov property

$$p(r_t, s_{t+1} \mid s_0, a_0, r_0, \ldots, s_t, a_t) = p(r_t, s_{t+1} \mid s_t, a_t)$$

(next state, expected reward) depend on (previous state, action)

Goal: solve the MDP by finding an optimal policy

1. Reward design
2. Bellman Equations
    a. state-value function
    b. action-value function
3. Policy: evaluation and improvement
4. Generalized Policy Iteration
    a. Policy Iteration
    b. Value iteration

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Cumulative reward is called a return:

$$G_t \triangleq R_t + R_{t+1} + R_{t+2} + ... + R_T$$

E.g.: reward in **chess** – value of taken opponent's piece

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Cumulative reward is called a return:

end of episode

$$\boxed{G_t} \triangleq \boxed{R_t} + R_{t+1} + R_{t+2} + \dots + R_{\boxed{T}}$$

immediate reward

E.g.: reward in **chess** – value of taken opponent's piece

E.g.: data center non-stop cooling system

- **S**tates – temperature measurements
- **A**ctions – different fans speed
- **R = 0** for exceeding temperature thresholds
- **R = +1** for each second system is cool

What could go wrong with such a design?

E.g.: data center non-stop cooling system

- **S**tates – temperature measurements
- **A**ctions – different fans speed
- **R = 0** for exceeding temperature thresholds
- **R = +1** for each second system is cool

What could go wrong with such a design?

Infinite return for non optimal behaviour!

$$G_t = 1 + 1 + 0 + 1 + 1 + 0 + ... = \sum_{t=1}^{\infty} R_t = \infty$$

E.g.: cleaning robot

- **S**tates – dust sensors, air
- **A**ctions – cleaning / rest / conditioning on or off
- **R = 100** for long tedious floor cleaning task done
- **R = 1** for turning air conditioning on-off
- Episode ends each day

What could go wrong with such a design?

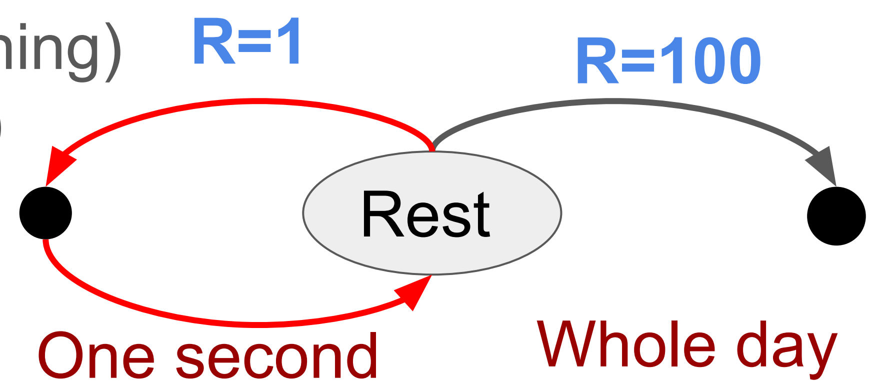
E.g.: cleaning robot

- **S**tates – dust sensors, air
- **A**ctions – cleaning / rest / conditioning on or off
- **R = 100** for long tedious floor cleaning task done
- **R = 1** for turning air conditioning on-off
- Episode ends each day

What could go wrong with such a design?

Reward(air) < Reward(cleaning)
Time(air) << Time(cleaning)

Positive feedback loop!



Reward discounting

Reward discounting

Get rid of infinite sum by discounting $0 \leq \gamma < 1$

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

discount factor

The same cake compared to today's one worth

- γ times less tomorrow
- γ^2 times less the day after tomorrow

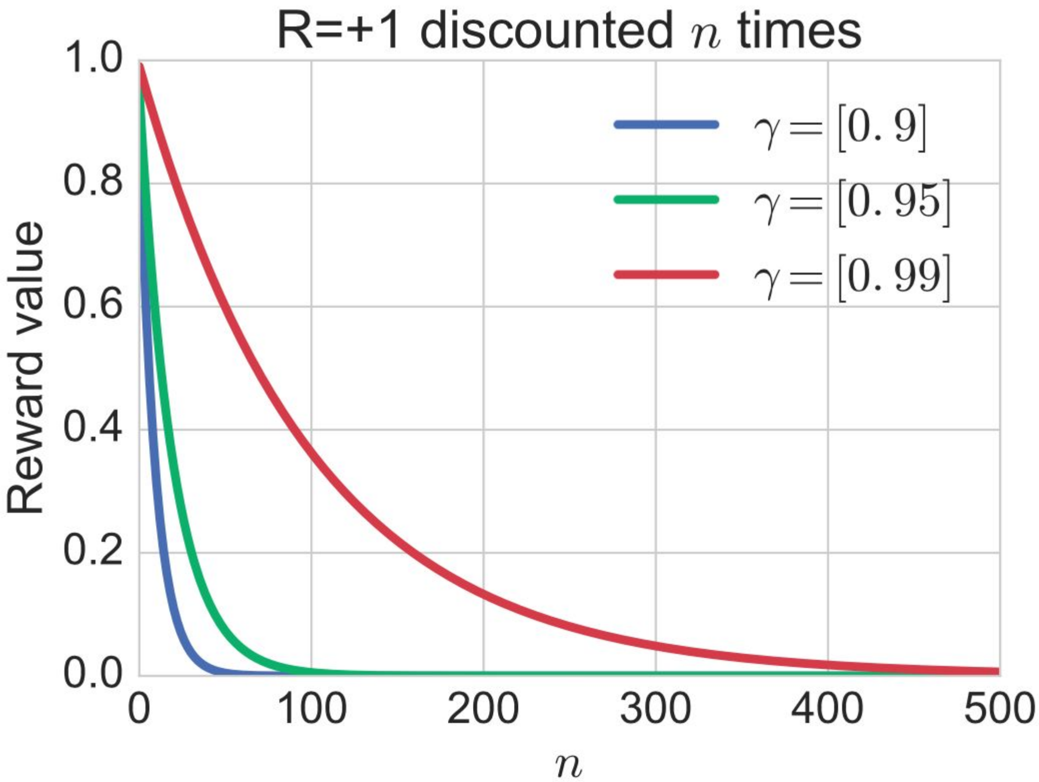
γ will eat it day by day

Discounting makes sums finite

Maximal return for **R = +1**

$$G_0 = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}$$

γ	0.9	0.95	0.99
$\frac{1}{1-\gamma}$	10	20	100



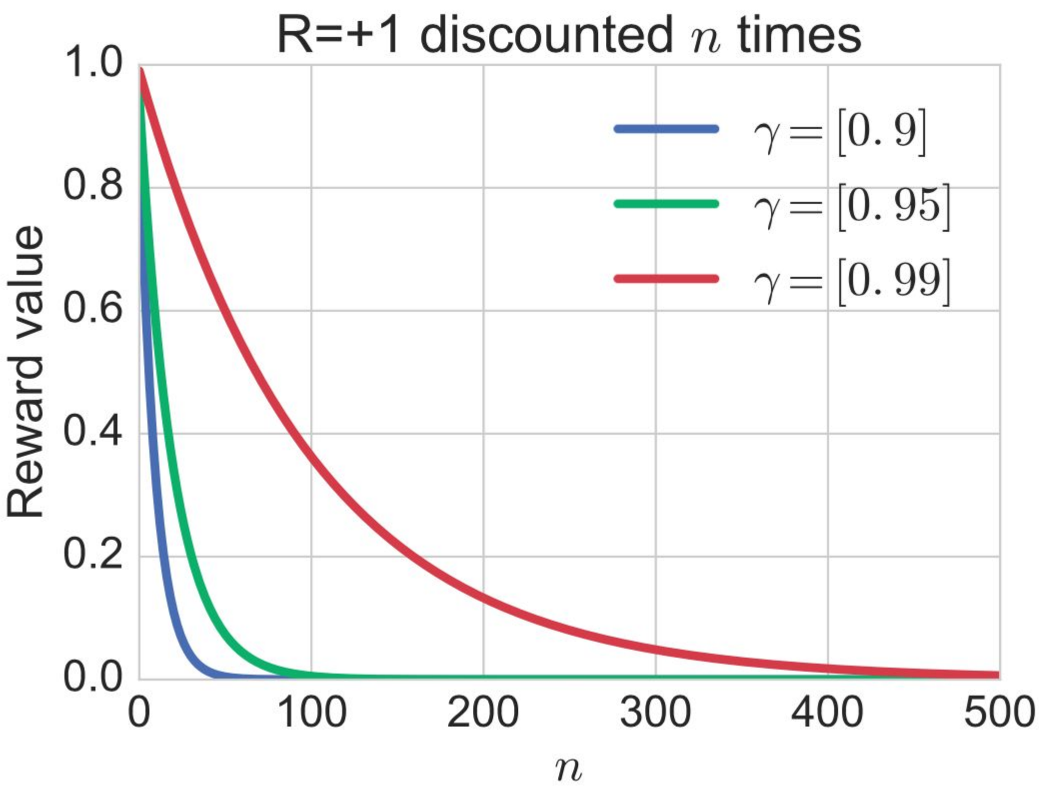
Discounting makes sums finite

Maximal return for **R = +1**

$$G_0 = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}$$

γ	0.9	0.95	0.99
$\frac{1}{1-\gamma}$	10	20	100

Any discounting changes optimisation task and its solution!



Discounting is inherent to humans

- Quasi-hyperbolic $f(t) = \beta\gamma^t$
- Hyperbolic discounting $f(t) = \dfrac{1}{1 + \beta t}$

Laibson, D. (1997). Golden eggs and hyperbolic discounting. The Quarterly Journal of Economics, 112(2), 443-478.

Discounting is inherent to humans

- Quasi-hyperbolic $f(t) = \beta\gamma^t$
- Hyperbolic discounting $f(t) = \dfrac{1}{1 + \beta t}$

Mathematical convenience

$$\begin{aligned} G_t &= R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \ldots) \\ &= \boxed{R_t + \gamma\, G_{t+1}} \end{aligned}$$

Remember this one!
We will need it later

Laibson, D. (1997). Golden eggs and hyperbolic discounting. The Quarterly Journal of Economics, 112(2), 443-478.

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) next state

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) next state

Action indirectly affects future rewards

But how long does this effect lasts?

$$
\begin{aligned}
G_0 &= R_0 + \gamma R_1 + \gamma^2 R_2 + \ldots + \gamma^T R_T \\
&= (1-\gamma) R_0 \\
&\quad + (1-\gamma)\gamma (R_0 + R_1) \\
&\quad + (1-\gamma)\gamma^2 (R_0 + R_1 + R_2) \\
&\quad \cdots \\
&\quad + \gamma^T \cdot \sum_{t=0}^{T} R_t
\end{aligned}
$$

G is expected return under stationary end-of-effect model

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) next state

Action indirectly affects future rewards

But how long does this effect lasts?

$$
\begin{aligned}
G_0 &= R_0 + \gamma R_1 + \gamma^2 R_2 + \ldots + \gamma^T R_T \\
&= \boxed{(1-\gamma)} R_0 \\
&\quad + \boxed{(1-\gamma)}\boxed{\gamma}(R_0 + R_1) \\
&\quad + \boxed{(1-\gamma)}\gamma^2(R_0 + R_1 + R_2) \\
&\quad \ldots \\
&\quad + \gamma^T \cdot \sum_{t=0}^{T} R_t
\end{aligned}
$$

"End of effect" probability

"Effect continuation" probability

G is expected return under stationary end-of-effect model

Reward design – don’t shift, reward for WHAT

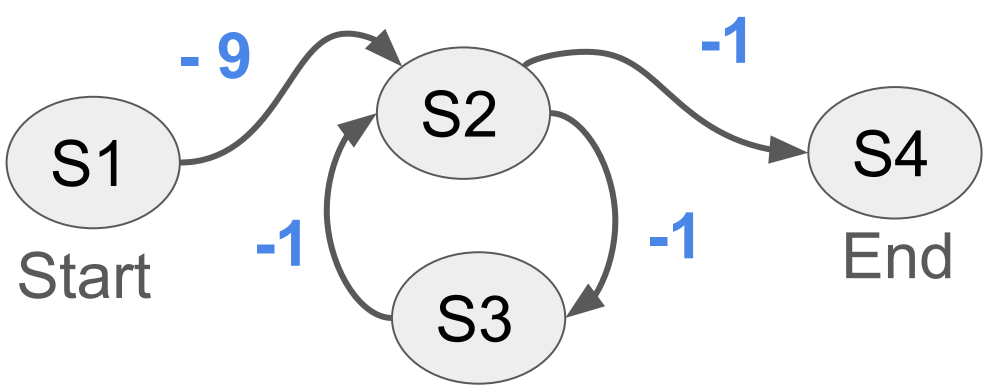
- E.g.: chess – value of taken opponent's piece
  - Problem: agent will not have a desire to win!
- E.g.: cleaning robot, **+100** (cleaning), **+0.1** (on-off)
  - Problem: agent will not bother cleaning the floor!

Reward design – don’t shift, reward for WHAT

- E.g.: chess – value of taken opponent's piece
  - Problem: agent will not have a desire to win!

- E.g.: cleaning robot, **+100** (cleaning), **+0.1** (on-off)
  - Problem: agent will not bother cleaning the floor!

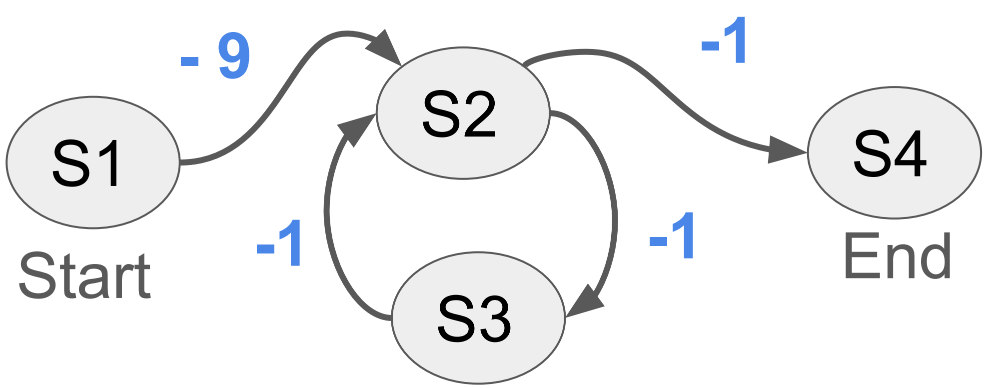
Take away: reward only for WHAT, but never for HOW



Reward design – don’t shift, reward for WHAT

- E.g.: chess – value of taken opponent's piece
    - Problem: agent will not have a desire to win!
- E.g.: cleaning robot, **+100** (cleaning), **+0.1** (on-off)
    - Problem: agent will not bother cleaning the floor!

Take away: reward only for WHAT, but never for HOW



Take away: do not subtract mean from rewards

Reward design – scaling, shaping

What transformations do not change optimal policy?

- Reward **scaling** – division by nonzero constant
  - May be useful in practise for approximate methods

Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In ICML (Vol. 99, pp. 278-287).

Reward design – scaling, shaping

What transformations do not change optimal policy?

- Reward **scaling** – division by nonzero constant
  - May be useful in practise for approximate methods
- Reward **shaping** – we could add to all rewards in MDP values of potential-based shaping function F(s, a, s') without changing an optimal policy:

$$F(s, a, s') = \gamma\Phi(s') - \Phi(s)$$

Intuition: when no discounting F adds as much as it subtracts from the total return

Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In ICML (Vol. 99, pp. 278-287).

Lecture plan

1. Reward design
2. Bellman Equations
   a. state-value function
   b. action-value function
3. Policy: evaluation and improvement
4. Generalized Policy Iteration
   a. Policy Iteration
   b. Value iteration

How to find optimal policy?

Dynamic programming!

Method to solve a complex problem by

- breaking it into small pieces
- until no more unsolved pieces
  - solve a single piece using solutions of previous pieces

DP equations lies at the heart of RL

It is essential to deeply understand them.

How to find optimal policy?

We know! Maximize cumulative discounted return!

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

How to find optimal policy?

We know! Maximize cumulative discounted return!

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

But policy and / or environment could be random!


Let get rid of randomness by taking expectation!

Equivalent variants of notation in RL

$$
\begin{aligned}
\mathbb{E}\left[G_0\right] &= \mathbb{E}\left[R_0 + \gamma R_1 + ... + \gamma^T R_T\right] \\
&= \mathbb{E}_{E,\pi_\theta}\left[\, G_0 \,\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\, G_0 \,\right] \\
&= \mathbb{E}\left[G_0 \mid \pi_\theta\right] \\
&= \mathbb{E}_{\substack{s_{0:T} \\ a_{0:T}}}\left[\, G_0 \,\right] \\
&= \mathbb{E}_{s_0}\left[\, \mathbb{E}_{a_0|s_0}\left[\, R_0 + \mathbb{E}_{s_1|s_0,a_0}\left[\, \mathbb{E}_{a_1|s_1}\left[\, \gamma R_1 + ... \,\right]\,\right]\,\right]\,\right] \\
&= \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim p_\theta}\left[\, \gamma^t R_t \,\right] \\
&= \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\, G(\tau) \,\right]
\end{aligned}
$$

Equivalent variants of notation in RL

$$
\begin{aligned}
\mathbb{E}\left[G_0\right] &= \mathbb{E}\left[R_0 + \gamma R_1 + \ldots + \gamma^T R_T\right] \\
&= \mathbb{E}_{E,\pi_\theta}\left[\, G_0 \,\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\, G_0 \,\right] \\
&= \mathbb{E}\left[G_0 \mid \pi_\theta\right] \\
&= \mathbb{E}_{\substack{s_{0:T} \\ a_{0:T}}}\left[\, G_0 \,\right] \\
&= \mathbb{E}_{s_0}\left[\, \mathbb{E}_{a_0|s_0}\left[\, R_0 + \mathbb{E}_{s_1|s_0,a_0}\left[\, \mathbb{E}_{a_1|s_1}\left[\, \gamma R_1 + \ldots \,\right]\,\right]\,\right]\,\right] \\
&= \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim p_\theta}\left[\, \gamma^t R_t \,\right] \\
&= \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\, G(\tau) \,\right]
\end{aligned}
$$

$$\tau \triangleq (s_0, a_0, s_1, \ldots, a_{T-1}, s_T)$$

$$p_\theta(\tau) = p(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t)\; p(s_{t+1}|s_t, a_t)$$

State-value function v(s)

v(s) is expected return conditional on state:

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

Intuition: value of following policy π from state s

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

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

Intuition: value of following policy π from state s

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

Environment stochasticity

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

Policy stochasticity

Intuition: value of following policy π from state s

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

Environment stochasticity

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

Policy stochasticity

By definition

Intuition: value of following policy π from state s

Bellman expectation equations

Bellman expectation equation for **v(s)**

Recursive definition of v(s) is an important concept in RL

$$v_\pi(s) = \sum_a \pi(a \mid s) \sum_{r,\, s'} p(r, s' \mid s, a) \left[r + \gamma v_\pi(s')\right]$$

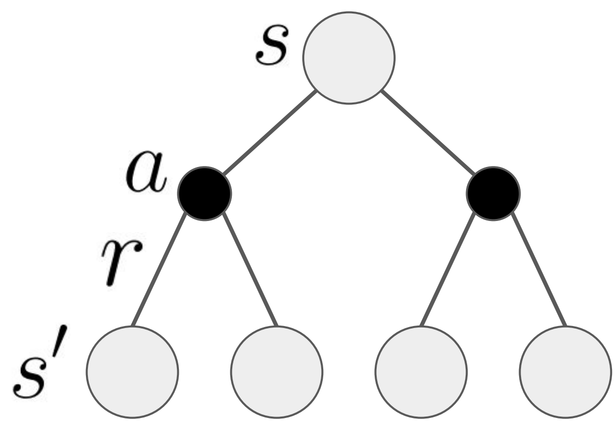
$$= \mathbb{E}_\pi \left[\, R_t + \gamma v_\pi(S_{t+1}) \mid S_t = s \,\right]$$

Bellman expectation equation for **v(s)**

Recursive definition of v(s) is an important concept in RL

$$v_\pi(s) = \sum_a \pi(a \mid s) \sum_{r,\, s'} p(r, s' \mid s, a) \left[r + \gamma v_\pi(s') \right]$$
$$= \mathbb{E}_\pi \left[R_t + \gamma v_\pi(S_{t+1}) \mid S_t = s \right]$$

Backup diagram



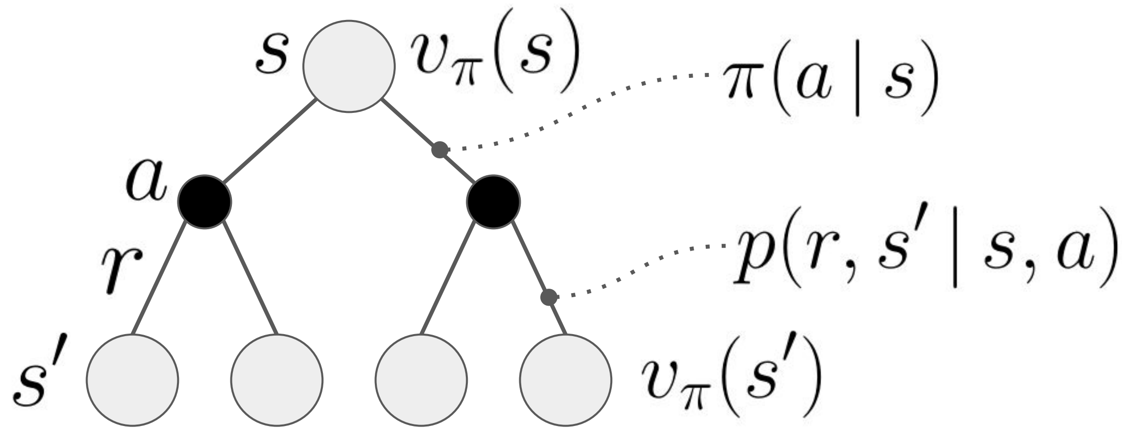
Bellman expectation equation for **v(s)**

Recursive definition of v(s) is an important concept in RL

$$v_\pi(s) = \sum_a \pi(a \mid s) \sum_{r,\, s'} p(r, s' \mid s, a) \left[r + \gamma v_\pi(s') \right]$$

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

Backup diagram



Action-value function q(s, a)

Is expected return conditional on state **and action**:

Intuition: value of following policy π after committing action **a** in state **s**

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

Action-value function q(s, a)

Is expected return conditional on state **and action**:

Intuition: value of following policy π after committing action **a** in state **s**

No policy stochasticity at first step

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[G_t \mid S_t = s, \boxed{A_t = a}\right] \\
&= \mathbb{E}_\pi\left[R_t + \gamma G_{t+1} \mid S_t = s, A_t = a\right] \\
&= \sum_{r,\,s'} p(r, s' \mid s, a)\Big[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\Big] \\
&= \sum_{r,\,s'} p(r, s' \mid s, a)\left[r + \gamma v_\pi(s')\right]
\end{aligned}
$$

Relations between v(s) and q(s,a)

We already know how to write q(s,a) in terms of v(s)

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

What about v(s) in terms of q(s,a)?

Relations between v(s) and q(s,a)

We already know how to write q(s,a) in terms of v(s)

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

What about v(s) in terms of q(s,a)?

$$\begin{aligned} v_\pi(s) &= \sum_a \pi(a \mid s) \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma v_\pi(s')\right] \\ &= \sum_a \pi(a \mid s) q_\pi(s,a) \end{aligned}$$

So, we could now write q(s, a) in terms of q(s,a)!

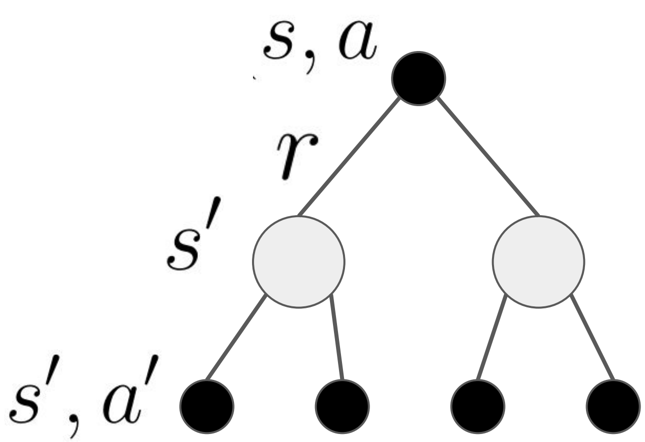
$$q_\pi(s,a) = \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma \sum_{a'} \pi(a' \mid s') q_\pi(s',a')\right]$$

Bellman expectation equation for **q(s,a)**

$$q_\pi(s,a) = \sum_{r,\,s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]$$

$$= \sum_{r,\,s'} p(r, s' \mid s, a)\Big[r + \gamma \sum_{a'} \pi(a' \mid s') q_\pi(s', a')\Big]$$

Backup diagram for q(s, a)

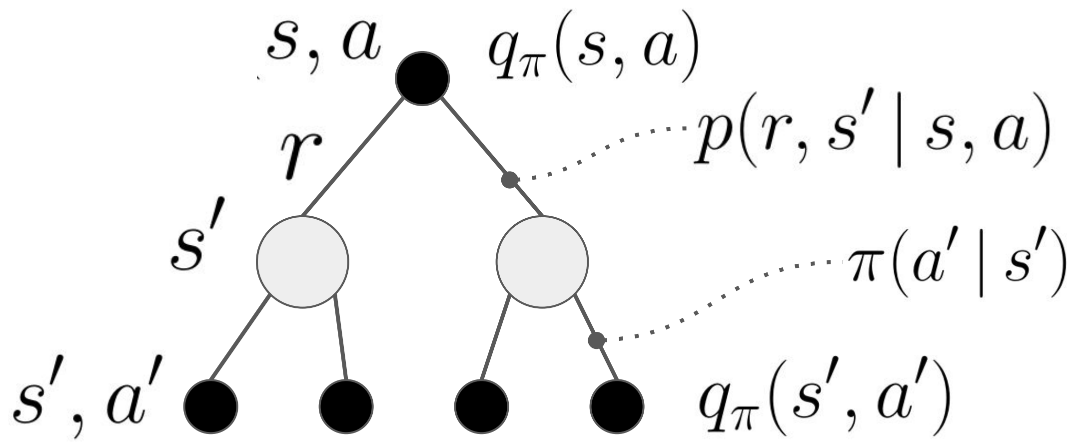


Bellman expectation equation for **q(s,a)**

$$q_\pi(s,a) = \sum_{r,\,s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]$$

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

Backup diagram for q(s, a)



What do we gonna do with value functions?

Already know

- Bellman equations – assess policy performance
- Return, value- and action-value functions

Want to find an optimal policy:

- optimal actions in each possible state

But how to know which policy is better?

How to compare them?

Optimal policy is the one with biggest v(s)

We could compare policies on the basis of v(s)

$$\pi \geq \pi' \quad \Leftrightarrow \quad v_\pi(s) \geq v_{\pi'}(s) \quad \forall\ s$$

Best policy π_* is better or equal to any other policy

Use optimal policy from s

$$\boxed{v_*(s) = \max_\pi v_\pi(s)}$$

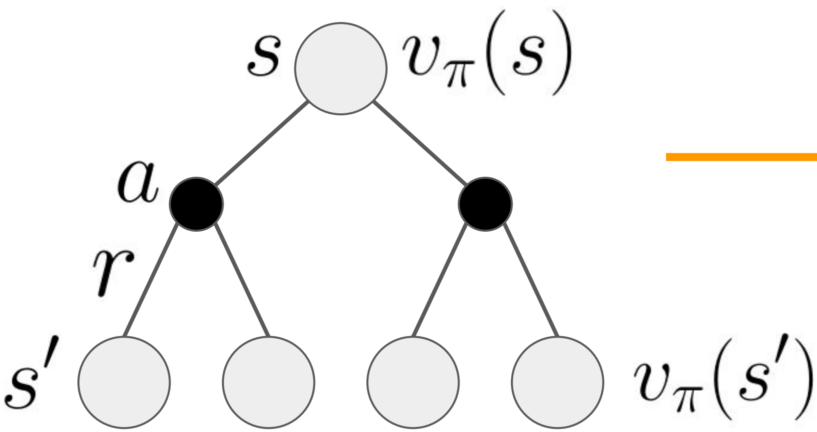
$$\boxed{q_*(s,a) = \max_\pi q_\pi(s,a)}$$

Commit action a, and **afterwards** use optimal policy

In any finite MDP there is always at least one deterministic optimal policy

Bellman optimality equations

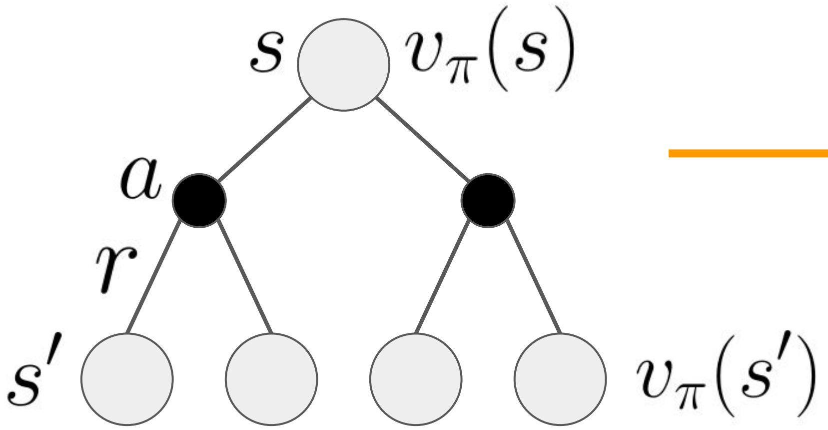
Bellman optimality equation for **v(s)**



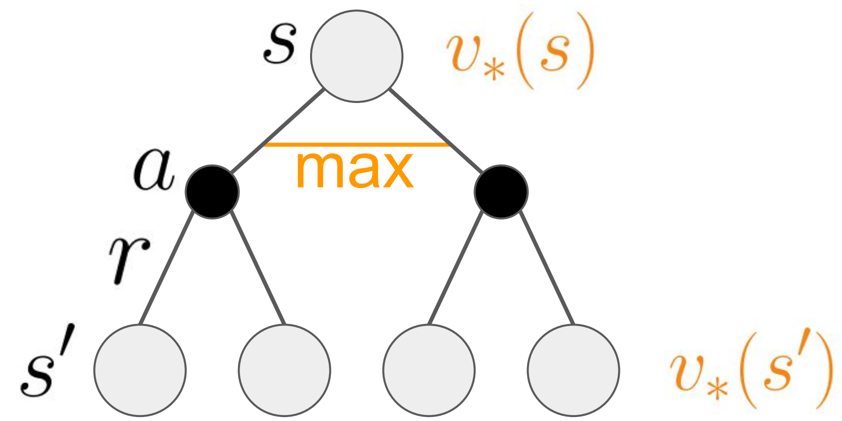
Bellman expectation equation for v(s)

Bellman optimality equation for $v_*(s)$

Bellman optimality equation for **v(s)**

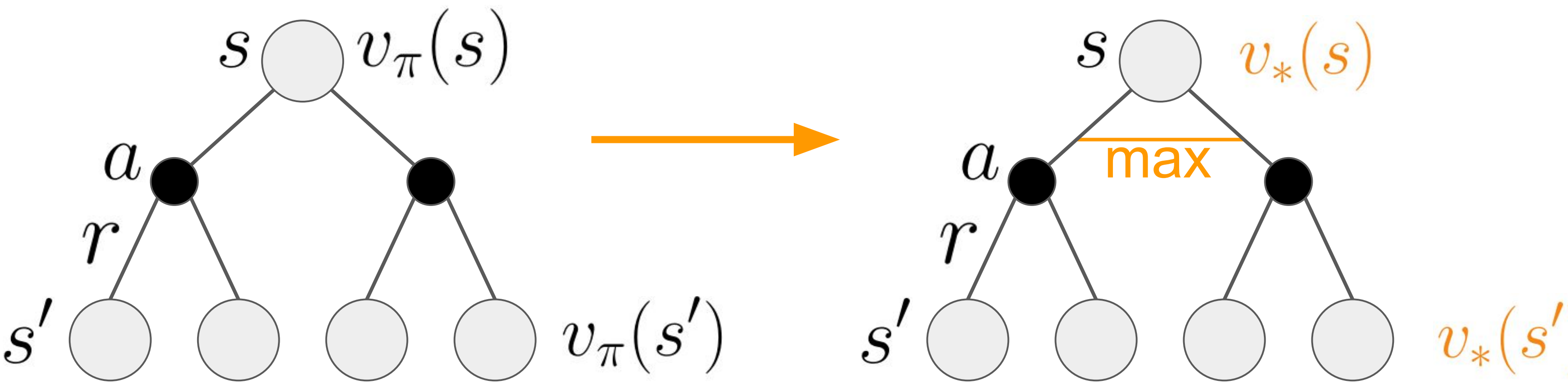


Bellman expectation equation for v(s)



Bellman optimality equation for $v_*(s)$

Bellman optimality equation for **v(s)**

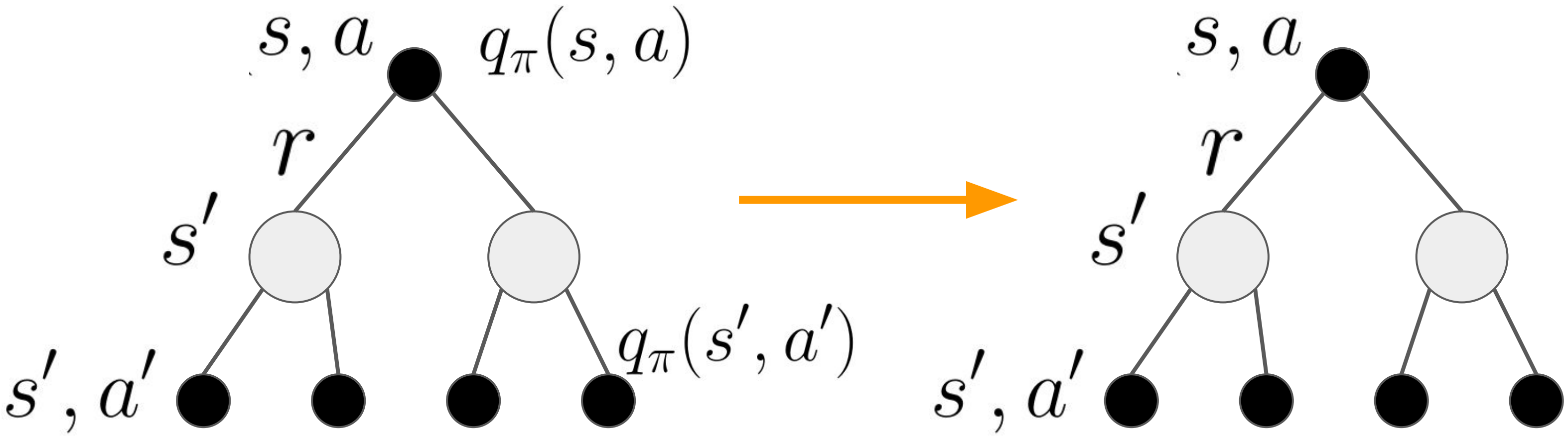


Bellman expectation equation for v(s)

Bellman optimality equation for $v_*(s)$

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

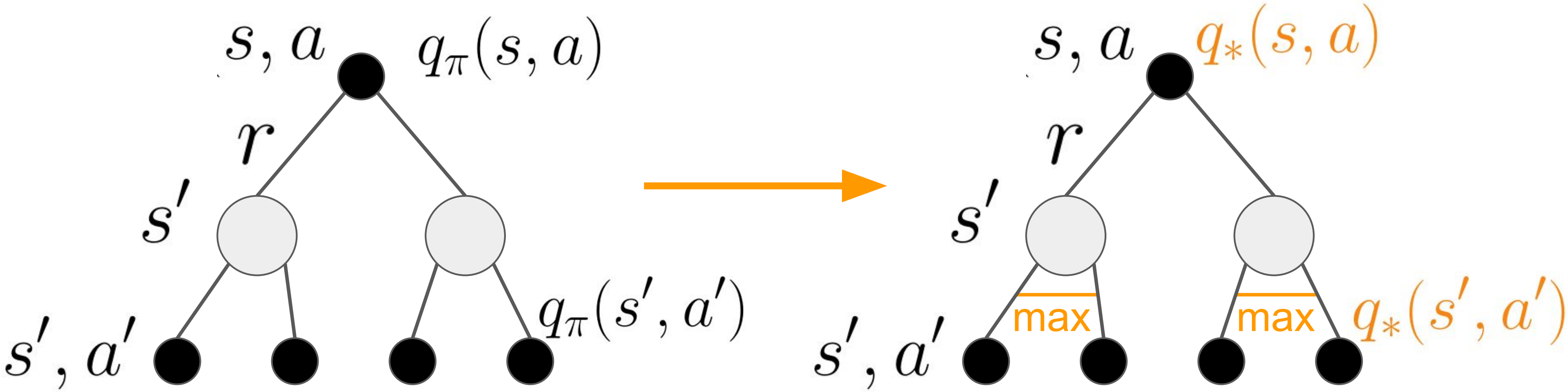
Bellman optimality equation for **q(s,a)**



Bellman expectation equation for q(s,a)

Bellman optimality equation for $q_*(s, a)$

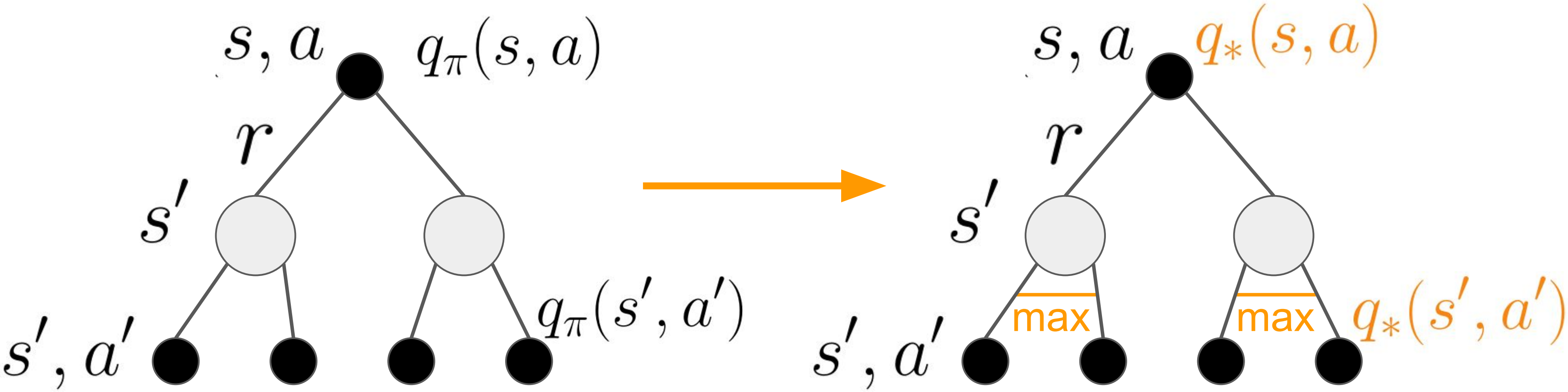
Bellman optimality equation for **q(s,a)**



Bellman expectation equation for q(s,a)

Bellman optimality equation for $q_*(s, a)$

Bellman optimality equation for **q(s,a)**



Bellman expectation equation for q(s,a)

Bellman optimality equation for $q_*(s, a)$

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

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

Bellman equations: operator view

$$[\mathcal{T}^\pi V](s) = \mathbb{E}_{r,s'|s,a=\pi(s)} \left[r + \gamma V(s') \right]$$

$$[\mathcal{T}^\pi Q](s,a) = \mathbb{E}_{r,s'|s,a} \left[r + \gamma \mathbb{E}_{a' \sim \pi(s')} \left[Q(s',a') \right] \right]$$

$$[\mathcal{T} V](s) = \max_a \mathbb{E}_{r,s'|s,a} \left[r + \gamma V(s') \right]$$

$$[\mathcal{T} Q](s,a) = \mathbb{E}_{r,s'|s,a} \left[r + \gamma \max_{a'} Q(s',a') \right]$$

Bellman equations: operator view

Bellman expectation equation for v(s)

$$[\mathcal{T}^\pi V](s) = \mathbb{E}_{r,s'|s,a=\pi(s)}\left[r + \gamma V(s')\right]$$

Bellman expectation equation for q(s,a)

$$[\mathcal{T}^\pi Q](s,a) = \mathbb{E}_{r,s'|s,a}\left[r + \gamma \mathbb{E}_{a'\sim\pi(s')}\left[Q(s',a')\right]\right]$$

Bellman optimality equation for $v_*(s)$

$$[\mathcal{T} V](s) = \max_a \mathbb{E}_{r,s'|s,a}\left[r + \gamma V(s')\right]$$

Bellman optimality equation for $q_*(s,a)$

$$[\mathcal{T} Q](s,a) = \mathbb{E}_{r,s'|s,a}\left[r + \gamma \max_{a'} Q(s',a')\right]$$

What's next?

Now we are equipped with heavy artillery of

- Bellman expectation equation for $v(s)$ and $q(s,a)$
- Bellman optimality equation for $v_*(s)$ and $q_*(s,a)$

That will be our toolkit for finding optimal policy

using dynamic programming!

Lecture plan

1. Reward design
2. Bellman Equations
    a. state-value function
    b. action-value function
3. Policy: evaluation and improvement
4. Generalized Policy Iteration
    a. Policy Iteration
    b. Value iteration

Policy evaluation

If you can't measure it, you can't improve it.
Peter Drucker

Policy evaluation: motivation

Policy evaluation is also called **prediction problem:**

- predict value function for a particular policy.

Bellman expectation equation

$$v_\pi(s) = \sum_a \pi(a \mid s) \sum_{r,\, s'} p(r, s' \mid s, a) \left[r + \gamma v_\pi(s') \right]$$
$$= \mathbb{E}_\pi \left[R_t + \gamma v_\pi(S_{t+1}) \mid S_t = s \right]$$

is basically a system of linear equations where

- # of unknowns = # of equations = # of states

Policy evaluation: algorithm

Input π, the policy to be evaluated
Initialize an array $V(s) = 0$, for all $s \in \mathcal{S}^+$
Repeat
$\quad \Delta \leftarrow 0$
$\quad$ For each $s \in \mathcal{S}$:
$\qquad v \leftarrow V(s)$
$\qquad \boxed{V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]}$
$\qquad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)
Output $V \approx v_\pi$

Bellman expectation equation for v(s)

Policy improvement

Policy improvement: an idea

Once we know what is v(s) for a particular policy

We could improve it by acting greedily w.r.t. v(s)!

$$\pi'(s) \leftarrow \arg\max_a \overbrace{\sum_{r,\, s'} p(r, s' \mid s, a)\left[r + \gamma v_\pi(s')\right]}^{q_\pi(s,\, a)}$$

This procedure is guaranteed to produce a better policy!

Policy improvement: an idea

Once we know what is v(s) for a particular policy

We could improve it by acting greedily w.r.t. v(s)!

$$\pi'(s) \leftarrow \arg\max_a \overbrace{\sum_{r,\,s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]}^{q_\pi(s,a)}$$

This procedure is guaranteed to produce a better policy!

if $\quad q_\pi(s, \pi'(s)) \geq v_\pi(s) \quad$ for all states

then $\quad v_{\pi'}(s) \geq v_\pi(s)$

meaning that $\quad \pi' \geq \pi$

Policy improvement: convergence

If new policy after improvement

$$\pi'(s) \leftarrow \arg\max_{a} \overbrace{\sum_{r,\, s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]}^{q_\pi(s,a)}$$

is the same as old one

$$\pi' = \pi \quad \rightarrow \quad v_{\pi'} = v_\pi$$

then it is optimal !

$$v_{\pi'}(s) = \max_{a} \sum_{r,\, s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]$$

Policy improvement: convergence

If new policy after improvement

$$\pi'(s) \leftarrow \arg\max_{a} \overbrace{\sum_{r,\, s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]}^{q_\pi(s,\,a)}$$

is the same as old one

$$\pi' = \pi \quad \rightarrow \quad v_{\pi'} = v_\pi$$

then it is optimal !

Bellman optimality equation

$$\boxed{v_{\pi'}(s) = \max_{a} \sum_{r,\, s'} p(r, s' \mid s, a)\,[r + \gamma v_\pi(s')]}$$

Determining optimal policy from $v_*(s)$, $q_*(s,a)$

If q* is known – how to recover the optimal policy?

$$\pi_*(s) \leftarrow \arg\max_a q_*(s, a)$$

If v* is known – how to recover the optimal policy?

Determining optimal policy from $v_*(s)$, $q_*(s,a)$

If q* is known – how to recover the optimal policy?

$$\pi_*(s) \leftarrow \arg\max_a q_*(s, a)$$

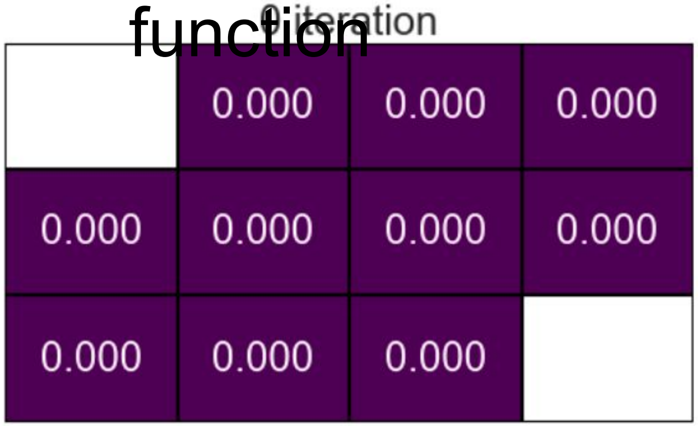
If v* is known – how to recover the optimal policy?

$$\pi_*(s) \leftarrow \arg\max_a \overbrace{\sum_{r,s'} \underline{p(r, s' \mid s, a)} \left[r + \gamma v_*(s')\right]}^{q_*(s,a)}$$

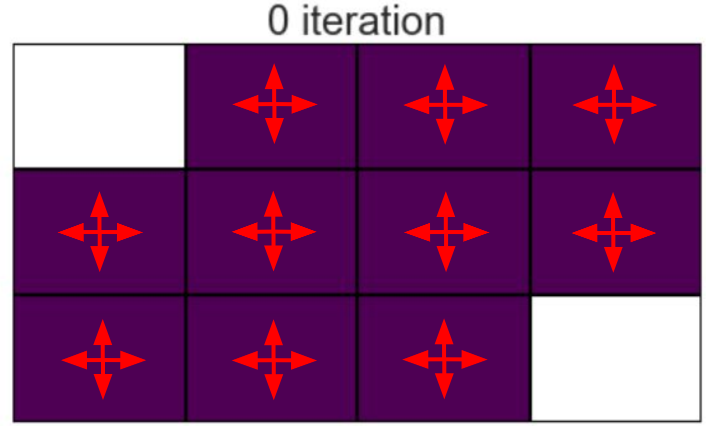
Unknown model dynamics → unable to recover optimal policy from v*

Precise evaluation is not needed

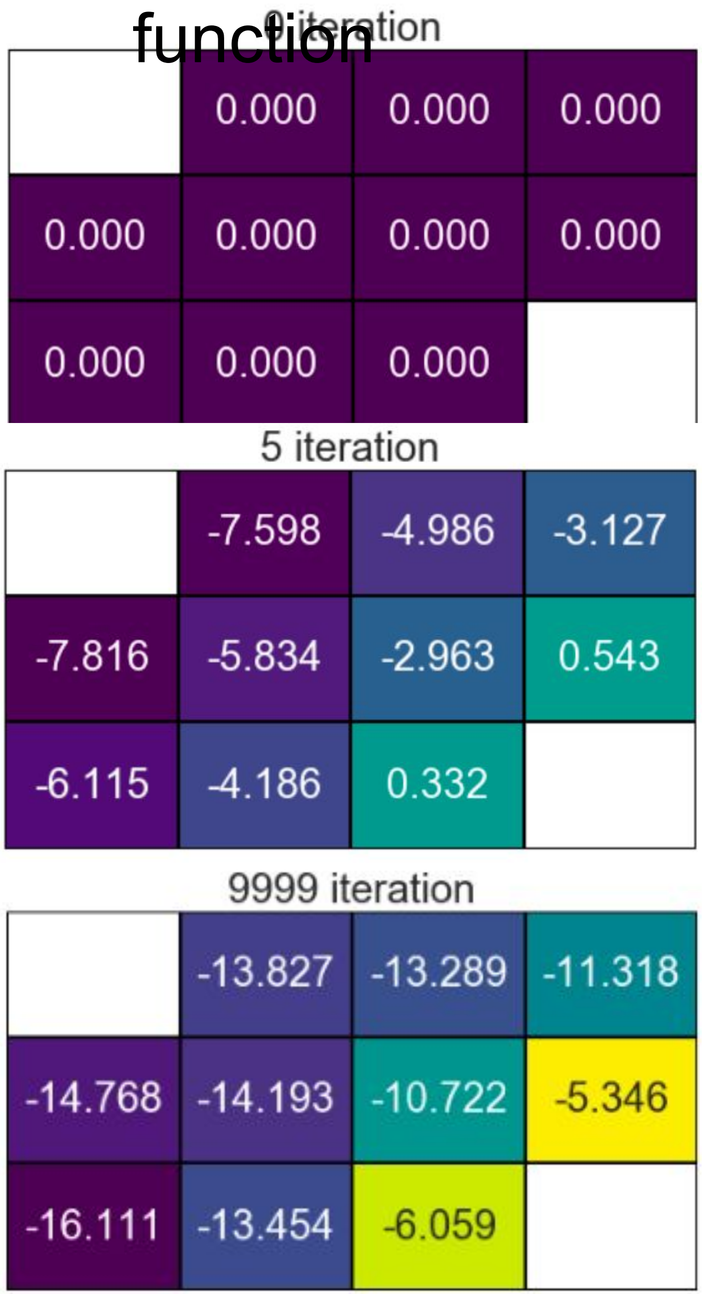
Value function



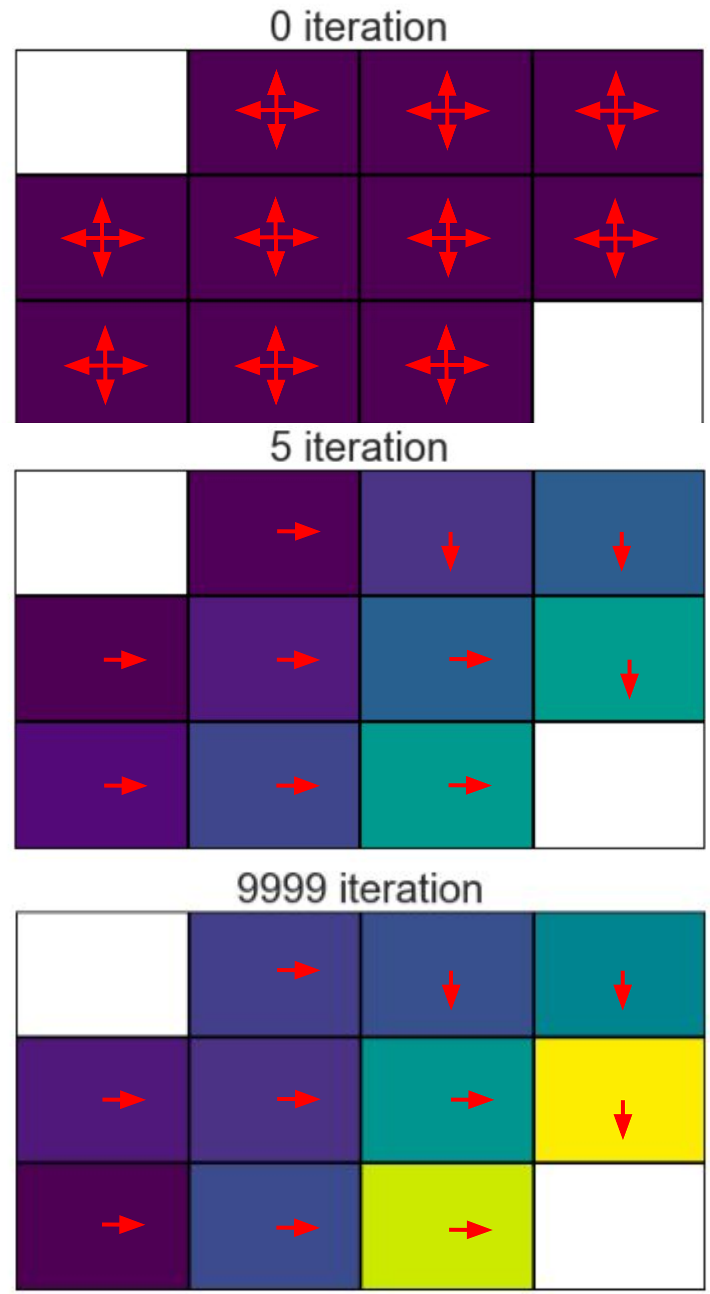
Greedy policy



Value function



Greedy policy



Roadmap

Now we know what is

- Policy evaluation (based on Bellman expectation eq)
- Policy improvement (based on Bellman optimality eq)

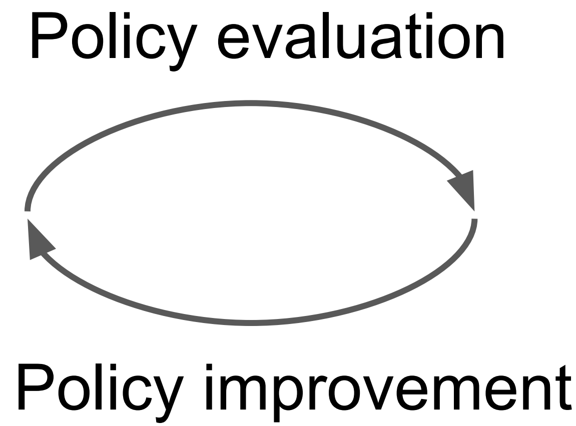
The finishing touches:

how to combine them to obtain optimal policy?

Lecture plan

1. Reward design
2. Bellman Equations
   a. state-value function
   b. action-value function
3. Policy: evaluation and improvement
4. Generalized Policy Iteration
   a. Policy Iteration
   b. Value iteration

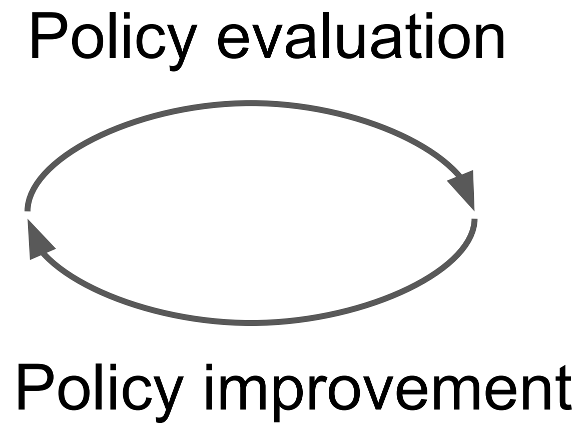
The idea of policy and value iterations



The idea of policy and value iterations

Generalized policy iteration

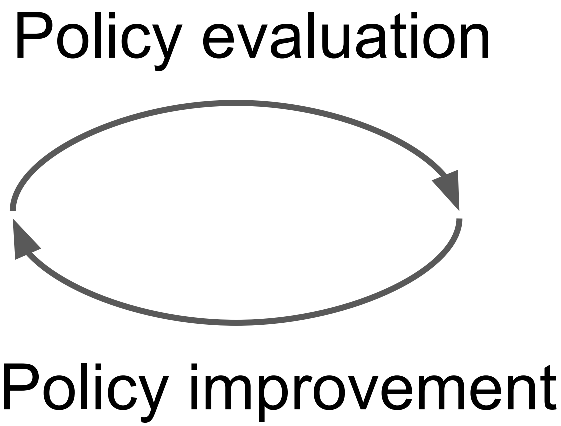
1. Evaluate given policy
2. Improve policy by acting greedily w.r.t. to its value function



The idea of policy and value iterations

Generalized policy iteration

1. Evaluate given policy
2. Improve policy by acting greedily w.r.t. to its value function



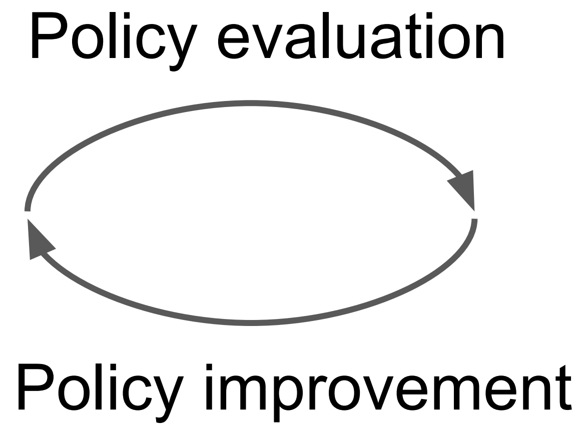
Robustness:

- No dependence on initialization
- No need in complete policy evaluation (states / converg.)
- No need in exhaustive update (states)
    - Example of update robustness:
        - Update only one state at a time
        - in a random direction
        - that is correct only in a expectation

The idea of policy and value iterations

Generalized policy iteration

1. Evaluate given policy
2. Improve policy by acting greedily w.r.t. to its value function



Policy iteration

1. Evaluate policy until convergence (with some tolerance)
2. Improve policy

Value iteration

1. Evaluate policy only with single iteration
2. Improve policy

Policy iteration

Policy iteration: scheme

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

2. Policy Evaluation
   Repeat
   $\quad \Delta \leftarrow 0$
   $\quad$ For each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad \boxed{V(s) \leftarrow \sum_{s',r} p(s',r|s,\pi(s))\big[r + \gamma V(s')\big]}$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

Bellman expectation equation for v(s)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \operatorname{argmax}_a \boxed{\sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]}$
   $\quad$ 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

q(s,a)

Value iteration

Value iteration

Initialize array V arbitrarily (e.g., $V(s) = 0$ for all $s \in \mathcal{S}^+$)

Bellman optimality equation for v(s)

Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $\boxed{V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]}$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

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

Value iteration (VI) vs. Policy iteration (PI)

- VI is faster per iteration – $O(|A||S|^2)$
- VI requires many iterations

- PI is slower per iteration – $O(|A||S|^2 + |S|^3)$
- PI requires few iterations

No silver bullet $\rightarrow$ experiment with # of steps spent in policy evaluation phase to find the best