# Lecture 2: Bellman operator, Banach's fixed point, solving MDPs

## SUMS 707 - Basic Reinforcement Learning

Gabriela Moisescu-Pareja and Viet Nguyen

McGill University, Mila

January 21, 2021

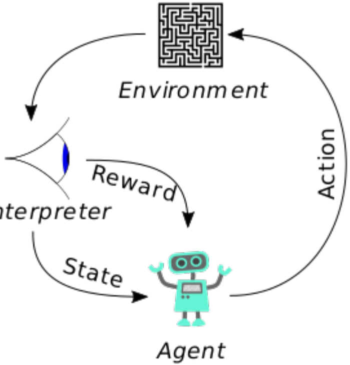
# Recap of last lecture

Last time we introduced:

- Markov Decision Processes
- Policies
- State value functions, action-value functions
- Optimality

# Recap of last lecture: RL, MDPs I

- Agent is in some state, performs actions
- Environment transitions agent to a state, gives a reward
- This interactive process is *modeled* through a Markov Decision Process (MDP)
- Assume the *Markov property*



# Recap of last lecture: RL, MDPs II

- $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}_0)$ is an MDP, where $\mathcal{S}$ are states, $\mathcal{A}$ are actions, and $\mathcal{P}_0$ is a transition kernel
- the transition kernel gives a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and a *state transition kernel* $\mathcal{P}$:

$$\mathbb{P}\left(s' | s, a\right) = \mathcal{P}(s, a, s')$$

- Agent at state $s_t$ takes action $a_t$, the environment transitions the agent to state $s_{t+1}$ and gives reward $r_{t+1}$ as $(s_{t+1}, r_{t+1}) \sim \mathcal{P}_0(\cdot | s_t, a_t)$
- Want to maximize expected return, i.e. the expectation of

$$\mathcal{R} = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$$

# Recap of last lecture: Policies I

- Policies formalize an agent's behavior
- A policy $\pi$ is a mapping $\pi : \mathcal{S} \to \Delta(\mathcal{A})$
- The agent in state $s_t$ samples an action $a_t$ according to $a_t \sim \pi(\cdot|s_t)$
- Deterministic policies: $a_t = \pi(s_t)$

Goal: Find the optimal policy, i.e. the policy that results in the maximum expected returns.

- How? $\rightarrow$ Need to quantify how "good" a state is under some policy

# Recap of last lecture: Value functions

- Given some policy, need to quantify the value of a state. Define value functions and action-value functions:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \middle| s_0 = s, \pi\right], \quad s \in \mathcal{S}$$

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \middle| s_0 = s, a_0 = a, \pi\right], \quad s \in \mathcal{S}, a \in \mathcal{A}$$

- Answers the questions "Given that I'm playing policy $\pi$, what is the value of being in state $s$?" and "Given that I'm playing policy $\pi$, what is the value of being in state $s$ and taking action $a$?"

# Recap of last lecture: Optimality

Define the optimal value and action-value functions:

$$V^*(s) = \max_\pi V^\pi(s)$$

$$Q^*(s, a) = \max_\pi Q^\pi(s, a)$$

- Intuitively, they specify the best performance in the MDP
- If we have $Q^*$, we can play the greedy policy $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a)$
- By playing $\pi^*$, you're getting the most rewards possible
- Pog

# Plan for today

- Bellman equations, Bellman optimality equations
- Banach fixed point theorem and contractions
- Solving MDPs with DP
    - Policy iteration
    - Value iteration

# Bellman Equations I

Let's look at our familiar ravioli:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \middle| s_0 = s, \pi\right]$$

There's a pickle here: this involves an infinite computation. Computers don't like infinite computations. Is there anything we can do?
Yes! We unpickle the pickle!

$$\begin{aligned} V^{\pi}(s) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \middle| s_0 = s, \pi\right] \\ &= \mathbb{E}\left[r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots \middle| s_0 = s, \pi\right] \\ &= \mathbb{E}\left[r_1 + \gamma V^{\pi}(s_1) \middle| s_0 = s, \pi\right] \end{aligned}$$

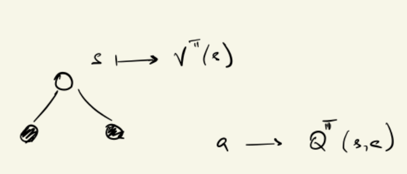
# Bellman Equations II

A similar decomposition gives us:

$$Q^{\pi}(s, a) = \mathbb{E}\left[r_1 + \gamma Q^{\pi}(s_1, a_1) | s_0 = s, a_0 = a, \pi\right]$$

These are things to keep in mind moving forward...

There seems to be some recursion going on...

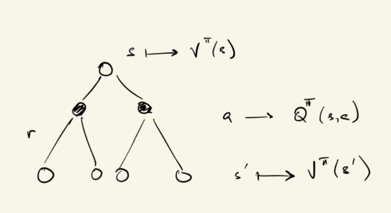



- In state $s$, we can take actions $a \in \mathcal{A}$ with probability $\pi(a|s)$
- By definition, we have that

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q^\pi(s,a)\right] \\ &= \sum_{a\in\mathcal{A}} \pi(a|s)\, Q^\pi(s,a) \end{aligned}$$

# Bellman Equations for $V^\pi$ II

One step look-ahead gives us:




- The environment moves you to state $s'$ with probability $\mathbb{P}\left(s'|s,a\right)$, gives you reward r.

$$
\begin{aligned}
V^\pi(s) &= \sum_{a\in\mathcal{A}} \pi(a|s)\, Q^\pi(s,a) \\
&= \sum_{a\in\mathcal{A}} \pi(a|s) \left( r_{(s,a)} + \gamma \sum_{s'\in\mathcal{S}} \mathbb{P}\left(s'|s,a\right) V^\pi(s') \right)
\end{aligned}
$$

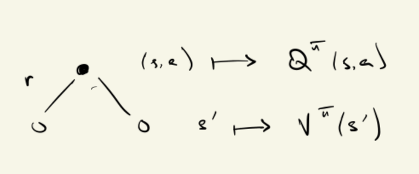
# Deterministic Bellman Equations for $V^\pi$

When $\pi$ is deterministic, this equation reduces to:

$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s, a) V^\pi(s')$$

where we used here the reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defined last time!

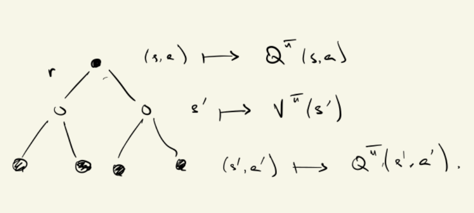
# Bellman Equations for $Q^\pi$ I




- Taking action $a$ in state $s$, we obtain a reward $r_{(s,a)}$ and transitions to $s'$ w.p. $\mathbb{P}(s'|s,a)$
- Thus, we have that

$$Q^\pi(s,a) = r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) V^\pi(s')$$

# Bellman Equations for $Q^\pi$ II




- In state $s'$, we choose action $a'$ according to $\pi$. But we already have a formuloi for $V^\pi$ from the previous slides!

$$
\begin{aligned}
Q^\pi(s,a) &= r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) V^\pi(s') \\
&= r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) \mathbb{E}_{a' \sim \pi(\cdot|s)} \left[Q^\pi(s', a')\right] \\
&= r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s', a')
\end{aligned}
$$

# Deterministic Bellman Equations for $Q^\pi$

$$Q^\pi(s,a) = r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s', a')$$

When $\pi$ is deterministic, this equation reduces to

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s,a) Q^\pi(s', \pi(s'))$$

# Bellman Operator I

Let's stare again real hard at the two *deterministic* equations we just wrote down:

$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s, a) V^\pi(s')$$

$$Q^\pi(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s, a) Q^\pi(s', \pi(s'))$$

What do you see?

# Bellman Operator II

$$V^{\pi}(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s, a) V^{\pi}(s')$$

$$Q^{\pi}(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(s'|s, a) Q^{\pi}(s', \pi(s'))$$

Yes! It does look like we can define an *operator* underlying $\pi$. Define the Bellman operator $T^{\pi} : (\mathcal{S} \to \mathbb{R}) \to (\mathcal{S} \to \mathbb{R})$ defined by its actions on $\mathcal{S}$ via *any* $V : \mathcal{S} \to \mathbb{R}$ in the following way:

$$(T^{\pi} V)(s) = r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s'|s, a\right) V(s')$$

# Bellman Operator III

In the same way, one can define the Bellman operator for functions of $\mathcal{S} \times \mathcal{A}$. Consider $T^\pi : (\mathcal{S} \times \mathcal{A} \to \mathbb{R}) \to (\mathcal{S} \times \mathcal{A} \to \mathbb{R})$ defined by:

$$(T^\pi Q)(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s,a\right) Q(s', \pi(s'))$$

# Bellman Operator IV

In this way, we can rewrite the Bellman equations as:

$$T^\pi V^\pi = V^\pi, \quad T^\pi Q^\pi = Q^\pi$$

Why do we care about this?

- This system is *linear*, $T^\pi$ is an affine linear operator.
- When $\gamma \in (0, 1)$, $T^\pi$ is a max-norm *contraction* and the fixed-point equation $T^\pi V = V$ has an *unique* solution.
- The unique solution is *exactly* $V^\pi$!
- Similarly for $Q^\pi$...

# Optimal Policy

Define a partial ordering over policies:

$$\pi \geq \pi' \text{ if } \forall s \in \mathcal{S}, V^{\pi}(s) \geq V^{\pi'}(s)$$

**Theorem**

*For any MDP, there is always an optimal policy $\pi^*$ (not necessarily unique). All optimal policies achieve the optimal value function $V^{\pi^*}(s) = V^*(s)$ and optimal action-value function $Q^{\pi^*}(s, a) = Q^*(s, a)$.*

There will always be a deterministic optimal policy, so if we know $Q^*$ we already have the optimal policy by playing greedily.

# Optimality, continued

Recall the definitions of the optimal value and action-value functions:

$$V^*(s) = \max_{\pi} V^{\pi}(s), \quad Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

As we stated previously, they can actually be rewritten in terms of each other:

$$V^*(s) = \sup_{a} Q^*(s,a)$$

and

$$Q^*(s,a) = r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' \middle| s,a\right) V^*(s')$$

# Exercise: Optimality equations

Exercise: We previously derived the Bellman equations for $V^\pi$ and $Q^\pi$. The idea now is to derive the Bellman equations for $V^*$ and $Q^*$ (we call these the Bellman optimality equations) by following the exact same process. This time, however, we know that $V^* = V^{\pi^*}$ where $\pi^*$ is an optimal deterministic policy.

Try to rewrite $V^*$ and $Q^*$ in terms of each other when you draw out the trees (like we did).

# Exercise: Hint

You should arrive at something that looks like this:

$$V^*(s) = \sup_a \left\{ R_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) V^*(s') \right\}$$

$$Q^*(s,a) = r_{(s,a)} + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}\left(s' | s, a\right) \sup_{a'} Q^*(s', a')$$

# Bellman optimality equations, why?

- Many decision-making methods can be viewed as ways of approximately solving the Bellman optimality equations
- DP methods like policy and value iterations are very closely related to the Bellman optimality equation
- In the RL setting where we don't know the underlying environment dynamics, many methods to finding a good policy can be understood as approximately solving the Bellman optimality equations, but using actual experienced transitions instead of the knowledge of the expected transitions
- We will look into this more closely in the near future

# Solving MDPs I

We are now ready to solve MDPs! There are various methods that exist to do this, and we list some of the fundamental methods:

- **Dynamic programming** (which we will focus on today)
- Monte Carlo methods
- Temporal-difference methods

# Solving MDPs II

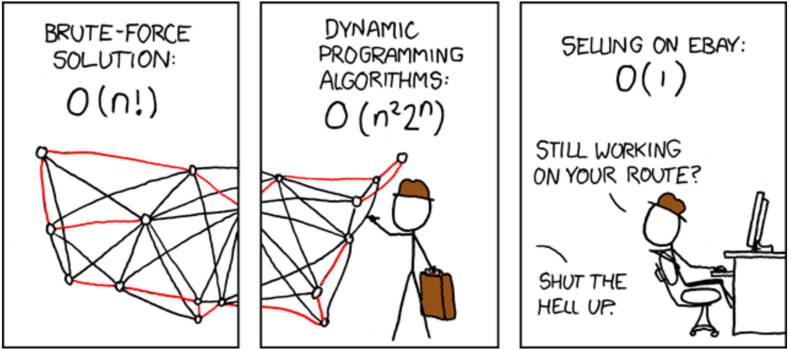
Of course, each method has its strengths and weaknesses.

- Dynamic programming:
  - Well-developed mathematically (in fact what we introduced in the previous section will come in handy now)
  - However, it requires a **full** description of the model of the environment (dynamics and rewards)
- Monte Carlo methods:
  - Don't require full model and are conceptually simple
  - However, they are not especially suited for step-by-step incremental computation
- Temporal-difference methods:
  - Don't require a full model
  - However, they are more complex to analyze (though this might have changed...).

# Solving MDPs III

Finally, the methods of course differ in terms of their computational efficiency and speed of convergence.

# Solving MDPs using dynamic programming I




A fun xkcd comic.

# Solving MDPs using dynamic programming II

We proceed by taking the dynamic programming (DP) approach to solving an MDP. The problem of finding $\pi^*$ given full specification of the MDP (model of the environment) is generally known as *planning*. There are two classical methods that we will go over:

- Policy iteration
- Value iteration

# Quick Dynamic Programming Overview I

DP allows us to solve complex problems by breaking them down into subproblems, solving the subproblems, and combining the solutions to those subproblems to arrive at a solution for the original problem. There two properties:

- Optimal substructure
  - Principle of optimality applies
  - Optimal solution can be decomposed into subproblems
- Overlapping subproblems
  - Subproblems recur many times
  - Solutions can be cached and reused

# Quick Dynamic Programming Overview II

As it turns out, MDPs satisfy this!

- Bellman equation gives recursive decomposition
- The value function stores and reuses solutions

# Quick Dynamic Programming Overview III

As we will see, these DP algorithms will be obtained by turning Bellman equations into update rules for improving approximations of the desired value functions.

# Policy evaluation I

Given a policy $\pi$, we would like to compute $V^{\pi}$. This is called *policy evaluation* (and sometimes called the prediction problem).

# Policy evaluation II

Recall

$$\begin{aligned} V^{\pi} &= \mathbb{E}\left[ r_{t+1} + \gamma V^{\pi}(s_{t+1}) \middle| s_t = s, \pi \right] \\ &= \sum_{a} \pi(s,a) \sum_{s'} \mathbb{P}\left(s' | s, a\right) \left[ r(s,a) + \gamma V^{\pi}(s') \right] \end{aligned}$$

# Policy evaluation III

Computing $V^\pi$ can be done iteratively and is known as *Iterative Policy Evaluation*. Consider a sequence of approximations $V_0, V_1, V_2, \ldots$

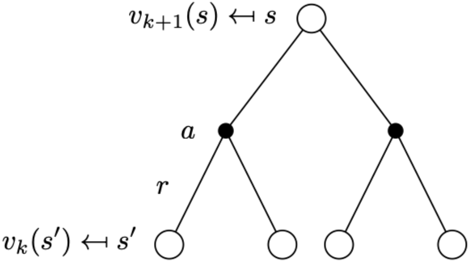
- Initial approximation $V_0$ is chosen arbirtrarily (though often set to 0)
- Each successive approximation is obtained by using the Bellman equation for $V^\pi$ as an update rule:

$$\begin{aligned} V_{k+1}(s) &= \mathbb{E}\left[ r_{t+1} + \gamma V_k(s_{t+1}) \middle| s_t = s, \pi \right] \\ &= \sum_a \pi(s,a) \sum_{s'} \mathbb{P}\left(s' | s, a\right) \left[ r(s,a) + \gamma V_k(s') \right] \end{aligned}$$

for all states $s \in S$.

We have that $V_k = V^\pi$ is a fixed point for this update rule thanks to the Bellman equation for $V^\pi$. The sequence $\{V_k\}$ can be shown, in general, to converge to $V^\pi$ as $k \to \infty$.

# Policy evaluation IV




We can understand the procedure graphically.

# Policy improvement I

Just now, we computed the value function given a policy $\pi$. This turns out to be useful to help us find *better policies*!

# Policy improvement II

Suppose we have computed the value function $V^\pi$ for a deterministic policy $\pi$.

- For some state $s$, we might like to know whether we should actually change the policy to deterministically choose another action $a \neq \pi(s)$.
- We know how good it is to follow the current policy from $s$ thanks to $V^\pi$.
- But we might wonder if it is better or worse to change to the new policy where we choose action $a$ instead of action $\pi(s)$ at state $s$.

# Policy improvement III

One way to answer this question is to consider choosing $a$ when in $s$ and following the existing policy $\pi$ thereafter. We can see how good this is by looking at at the state-action value function:

$$\begin{aligned} Q^{\pi}(s, a) &= \mathbb{E}\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) \middle| s_t = s, a_t = a, \pi\right] \\ &= \sum_{s'} \mathbb{P}\left(s' | s, a\right) \left[r(s, a) + \gamma V^{\pi}(s')\right] \end{aligned}$$

Is this greater than or less than $V^{\pi}$? If it is greater, it would be better to select $a$ and then follow $\pi$ than to follow $\pi$ the entire time. This key idea relies on the following *Policy Improvement Theorem*

# Policy improvement IV

**Theorem (Policy Improvement Theorem)**

*Choose some stationary policy $\pi_0$ and let $\pi$ be greedy w.r.t. $V^{\pi_0}$ : $T^\pi V^{\pi_0} = T^* V^{\pi_0}$. Then $V^\pi \geq V^{\pi_0}$, i.e. $\pi$ is an improvement upon $\pi_0$.*
*In particular, if $T^* V^{\pi_0}(s) > V^{\pi_0}(s)$ for some state $s$, then $\pi$ strictly improves upon $\pi_0$ at $s$, so*

$$V^\pi(s) > V^{\pi_0}(s).$$

*On the other hand, when $T^* V^{\pi_0} = V^{\pi_0}$ then $\pi_0$ is an optimal policy.*

The proof is omitted for now, but we will get to it next class.

# Policy iteration I

*Policy iteration* is an algorithm for the *control problem*. At a high-level, policy iteration performs the following steps:

1. Policy evaluation: value function is computed for the currently policy
2. Policy improvement: the policy is made greedy w.r.t. the value function
3. Repeat steps 1, 2 until convergence to optimal policy.

# Policy iteration II

Here we offer a run-down:

- We start with a deterministic policy $\pi$, where at state $s$ we act according to $a = \pi(s)$
- We can improve the policy by acting greedily

$$\pi'(s) = \arg\max_{a \in A} Q^{\pi}(s, a)$$

- This improves the value from any state $s$ over one step

$$Q^{\pi}(s, \pi'(s)) = \max_{a \in A} Q^{\pi}(s, a) \geq Q^{\pi}(s, \pi(s)) = V^{\pi}(s)$$

# Policy iteration III

- This improves the value function, $V^{\pi'}(s) \geq V^{\pi}(s)$

$$
\begin{aligned}
V^{\pi}(s) &\leq Q^{\pi}(s, \pi'(s)) \\
&= \mathbb{E}\left[ r_{t+1} + \gamma V^{\pi}(s_{t+1}) \middle| s_t = s, \pi' \right] \\
&\leq \mathbb{E}\left[ r_{t+1} + \gamma Q^{\pi}(s_{t+1}, \pi'(s_{t+1})) \middle| s_t = s, \pi' \right] \\
&\leq \mathbb{E}\left[ r_{t+1} + \gamma r_{t+2} + \gamma^2 Q^{\pi}(s_{t+2}, \pi'(s_{t+2})) \middle| s_t = s, \pi' \right] \\
&\leq \mathbb{E}\left[ r_{t+1} + \gamma r_{t+2} + \ldots \middle| s_t = s \right] \\
&= V^{\pi'}(s)
\end{aligned}
$$

# Policy iteration IV

- If improvements stop,

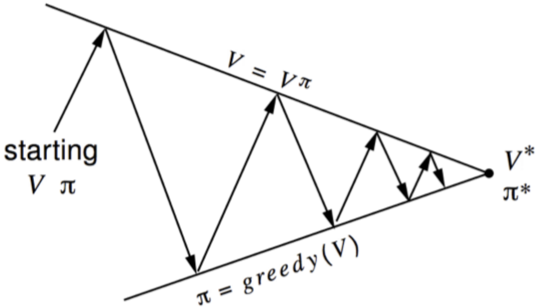
$$Q^{\pi}(s, \pi'(s)) = \max_{a \in A} Q^{\pi}(s, a) = Q^{\pi}(s, \pi(s)) = V^{\pi}(s)$$

- Then the Bellman optimality equation has been satisfied

$$V^{\pi}(s) = \max_{a \in A} Q^{\pi}(s, a)$$

Therefore $V^{\pi}(s) = V^{*}(s)$ for all states $s$ and $\pi$ is an optimal policy.

# Policy iteration V



# Value iteration I

Idea: Optimize value function directly and then induce a policy. This means that unlike in policy iteration, we don't need to go back and forth between value functions and policies.

# Value iteration II

We want to compute $V^*$ directly, and it can be done through an iterative method known as *ValueIteration*. Consider a sequence of approximations $V_0, V_1, V_2, \ldots$

- Initial approximation $V_0$ is chosen arbitrarily.
- Each successive approximation is obtained by

$$\begin{aligned} V_{k+1}(s) &= \max_{a \in A} \mathbb{E}\left[r_{t+1} + \gamma V_k(s_{t+1}) | s_t = s, a_t = a\right] \\ &= \max_{a \in A} \sum_{s'} \mathbb{P}\left(s' | s, a\right) \left[r(s, a) + \gamma V_k(s')\right] \end{aligned}$$

  for all $s \in S$.

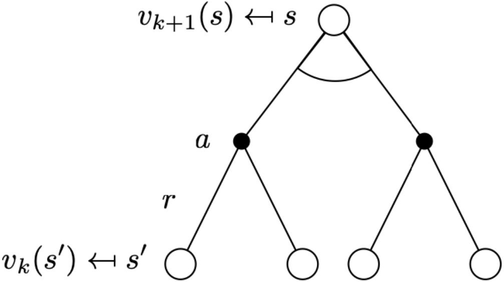
And $\{V_k\}$ can be shown to converge to $V^*$ under the same conditions that guarantee the existence of $V^*$.

# Value iteration III

On the convergence:

- We have that the Bellman optimality operator $T^*$ has a unique fixed point
- $V^*$ is a fixed point of $T^*$ by the Bellman optimality equation
- By the Banach fixed point theorem, value iteration converges to $V^*$ at a geometric rate.

# Value iteration IV




We can understand the procedure graphically.

# Important thingies

- MDPs are perfectly suited to DP given by the Bellman equations.
- DP methods such as PI and VI require a full model and are especially good for finite MDPs of small to moderate size, but other methods are necessary to tackle more complex RL problems
- Convergence of the PI and VI follow from the Banach fixed point theorem.

# Next lecture

What do we do when we don't know the dynamics of the environment?

- Model-free prediction and control
- TD and MC methods, Q-learning
- (maybe) Policy improvement theorem proof

# References


- Reinforcement Learning: Theory and Algorithms by Alekh Agarwal, Nan Jiang, Sham M. Kakade
- Algorithms for Reinforcement Learning by Csaba Szepesvári
- Reinforcement Learning: An Introduction by Andrew Barto and Richard S. Sutton
- "Introduction to Reinforcement Learning" Lectures by David Silver
- "CS 598 - Statistical Reinforcement Learning" Notes by Nan Jiang