Solving Known MDPs

Andrew Perrault
perrault.17@osu.edu

CSE 5539

Recap: MDPs

Markov decision processes (MDPs) formalize the problem of an agent interacting with an environment. What pieces do they have?

MDP formalism: states, action, transition function, reward function, discount factor, start state distribution
- **Markov property**: only the current state matters to next state and reward
- Goal: find policy (mapping of states to actions) that maximizes **expected discounted return** from the start state

Solving known MDPs is much easier than unknown MDPs
- Known MDPs with terminal states: **backwards induction**
- Unknown MDPs: not generally solvable, may require exponential time to even try all actions from all states
- This class: known MDPs without goal state

This lecture

We want to be able to find optimal policies in MDPs

This will be the basis for learning from experience

A few key theorems

We want to cover a few theorems that will simplify how we think about MDPs

1. For any fixed policy, there is an **optimal action** (or set of equally good actions) for each state (while keeping the rest of the policy fixed)
2. There is always a **deterministic optimal policy**, but there may not be a stochastic optimal policy (i.e., collapse to Dirac distribution may be necessary)
3. Finding the optimal policy is a **convex optimization problem** for a known MDP (and has an associated polynomial time algorithm)

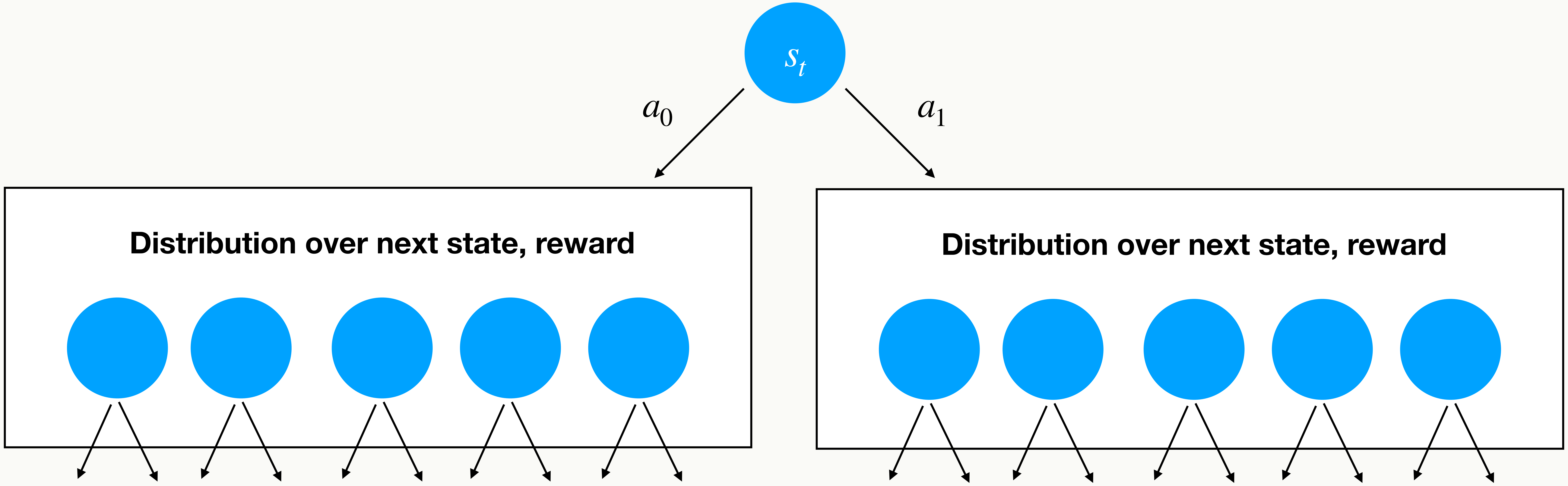
How do prove this stuff?

Key idea: exploit Markov property

Suppose we find ourselves in state s and have a fixed policy $\pi : S \to \Delta A$ and we want to figure out what the best action to take is (assuming we follow π from that point onward)

What is an action? *An action is just a probability distribution where the sample is over* (next state, next reward)

Action visualization: which is better?


s_t
a_0
a_1
Distribution over next state, reward
Distribution over next state, reward

For each action, we get a different distribution over immediate reward r_{t+1} (i.e., the reward associated with the transition (s_t, a_t, s_{t+1})) and over next state s_{t+1}

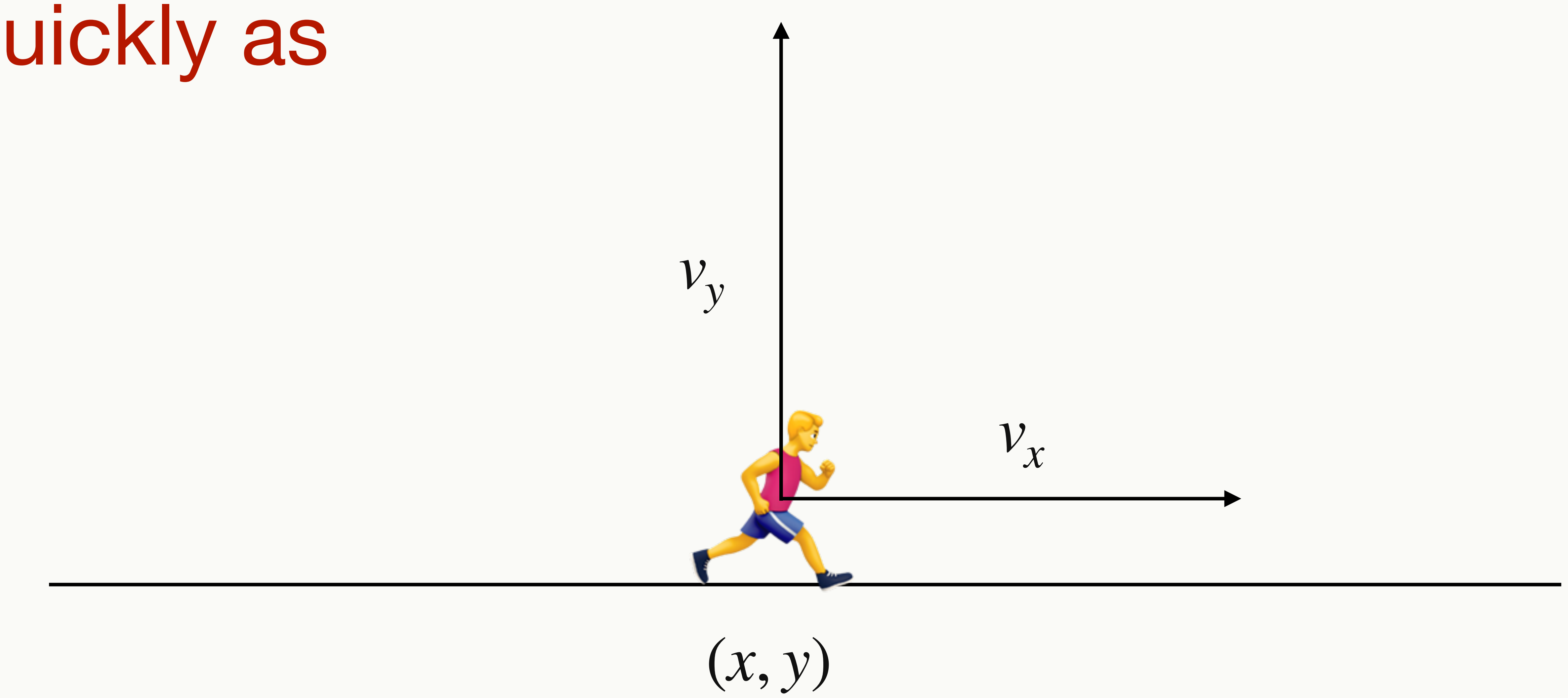
The better action is better because it gives us **more immediate reward (better r_{t+1}) and/or sets us up to get more reward in the future (better s_{t+1}). γ (discount factor) tells us how to balance these**

Exercise: MDP design for locomotion

Goal: move 2D robot to the right as quickly as possible

Actions:

- a_0: dive forward
- a_1: run forward



Two options for MDP state and reward: velocity-centric and position-centric

Exercise: locomotion MDPs

Which is better?

Position-centric MDP: state (x, y)

- $x \in \mathbb{R}$: x position—how far right is the robot?
- $y \in \mathbb{R}$: y position—how far up is the robot?
- Reward: two options

1. Reward = x (intuition: reward is average position over a trajectory)
2. Reward = x at time T and zero otherwise (intuition: maximize distance covered in fixed time)

Velocity-centric MDP: state (v_x, v_y)

- $v_x \in \mathbb{R}$: velocity to the right (negative → to the left)
- $v_y \in \mathbb{R}$: velocity up (negative → down)
- Reward = v_x

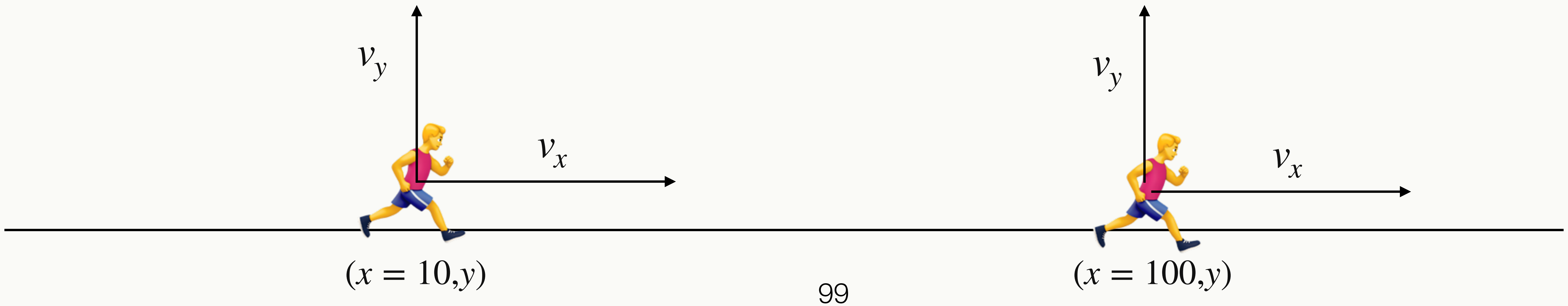
Biggest issue: generalization

Velocity-centric has better generalization

$x = 10$ and $x = 100$ look like different states, but they are not (in this example): the optimal action should not depend on absolute position
→ Much harder to learn in state-centric because the optimal action must be learned for each absolute position

Even worse, the optimal action could depend on velocity (stability) which we don't have access to

Goal: state should contain everything that is needed for optimal action and nothing else!



Reward design #1

Reward = v_x is nice

We receive reward every step (dense)—easier for us to know whether we are doing things well or badly (don't have to wait)

$$\sum_{t=0}^{T} v_x(s_t) = ?$$

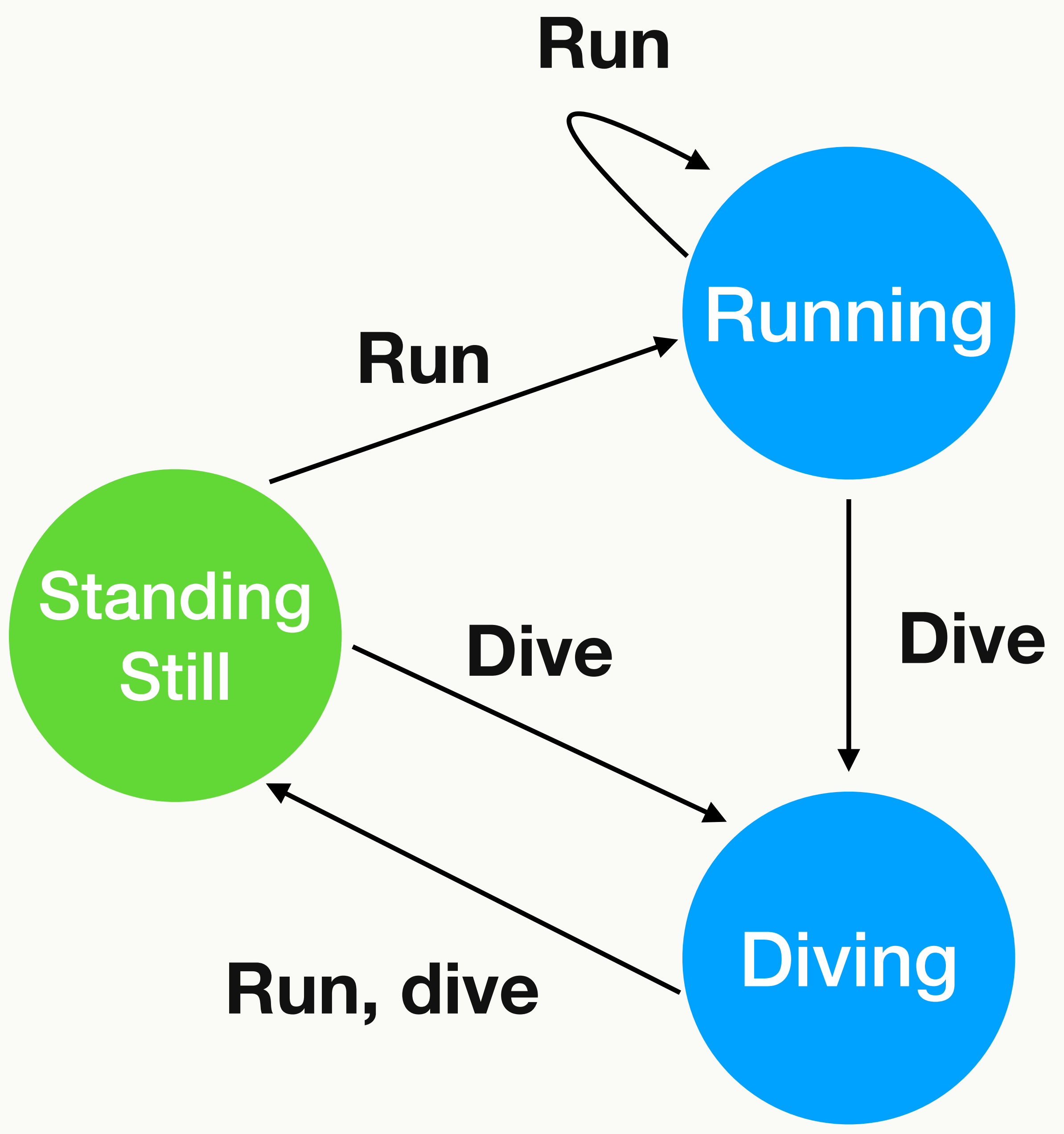
$\sum_{t=0}^{T} v_x(s_t) = x(s_t)$: telescoping reward! We maximize the x of the final state without storing it explicitly

Reward design #2

Consider reward = x at time T and zero otherwise
(intuition: maximize distance covered in fixed time)
- This a sparse reward (mostly zero, which is bad)
- What else is weird about it? Hint: T is a constant, what happens at $T-1$?

Because only position at T matters, the robot may do something like dive at time $T-1$ to get a bit more distance
- This is called a **horizon effect**, e.g., it's optimal to expend all resources for 1% more distance at T
- Horizon effects aren't necessarily bad: good if you want to win a race!

Exercise: locomotion MDP



Deterministic, infinite horizon MDP

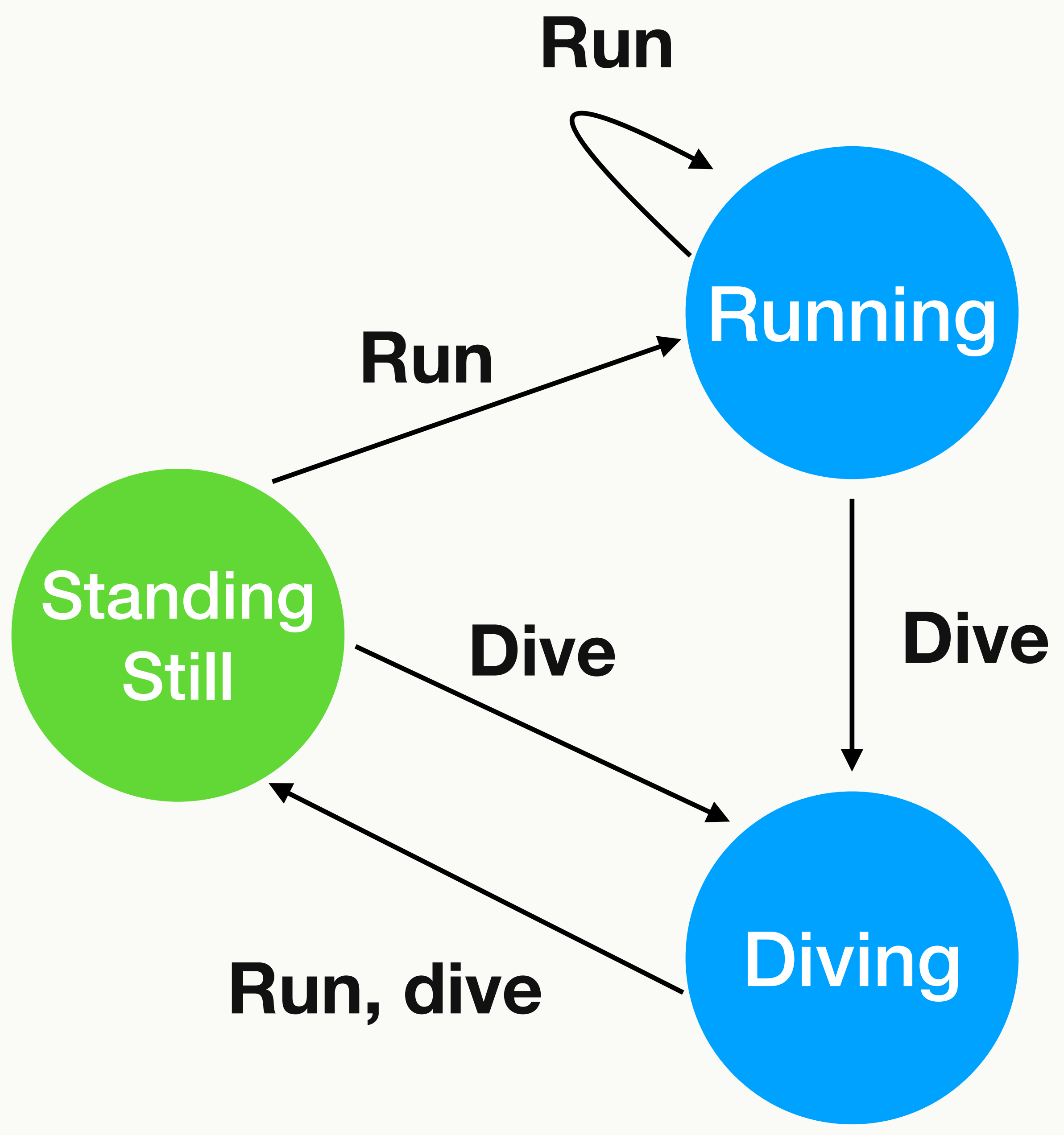
Reward = velocity of the state you are **entering**

- v_x(standing still) = 0
- v_x(running) = 1
- v_x(diving) = 1.5

Harder to solve than box K, can't do backwards induction from goal

Q1: How many deterministic policies are there?
Q2: What policy maximizes average reward?

Policies in locomotion MDP



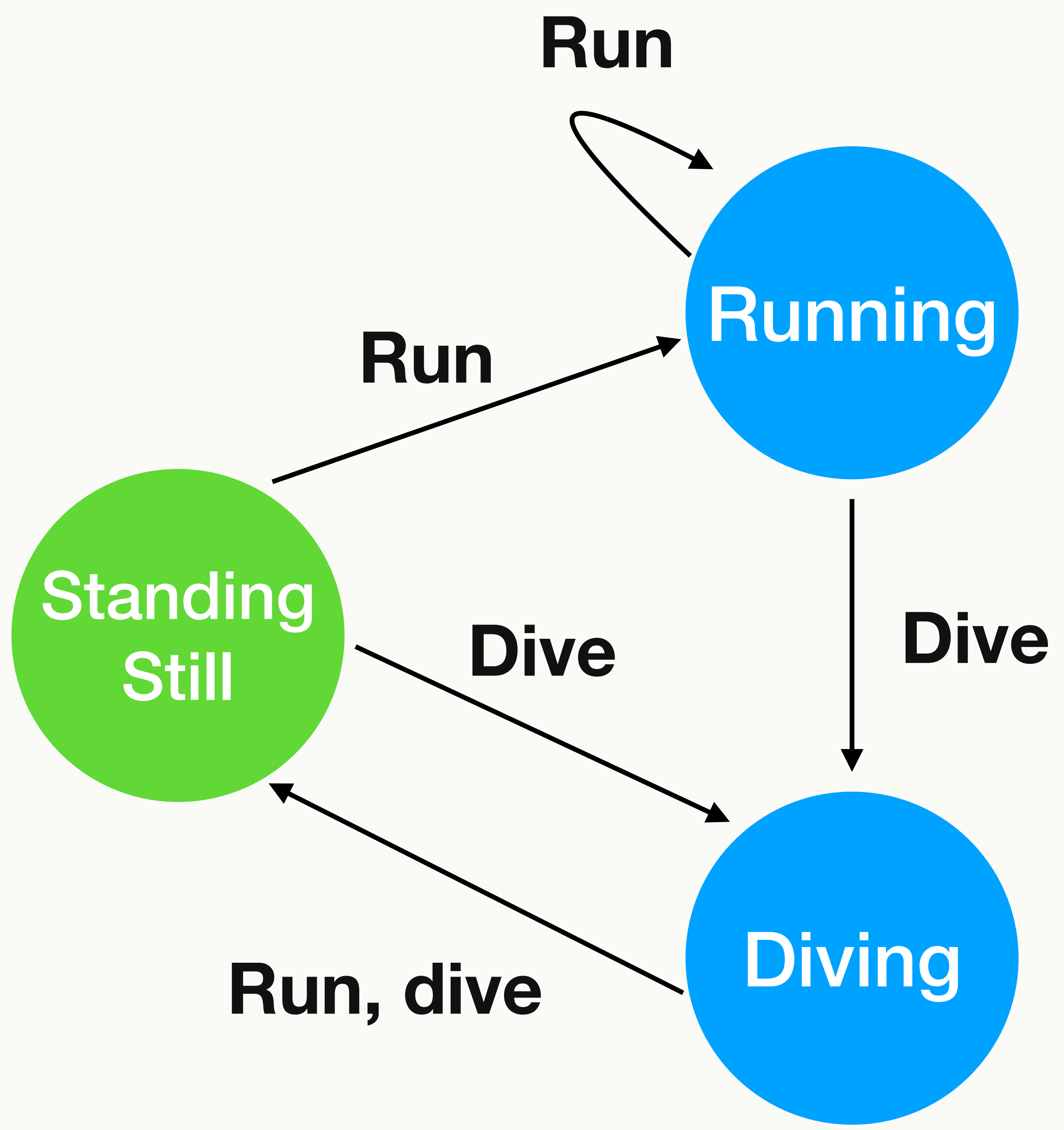
We only have two choices to make:
- Should we run or dive from Standing Still?
- Should we run or dive from Running?
- Choice in Diving doesn't matter

So 2 x 2 = 4 possible policies

How many are distinct?

3 are distinct:
- Run forever
- Dive forever (*choice in run doesn't matter*)
- Run, dive, x, run, dive, x

Which locomotion policy is best (average reward)?



Run forever $\rightarrow \{1,1,1,\ldots\}$ Avg. reward = 1

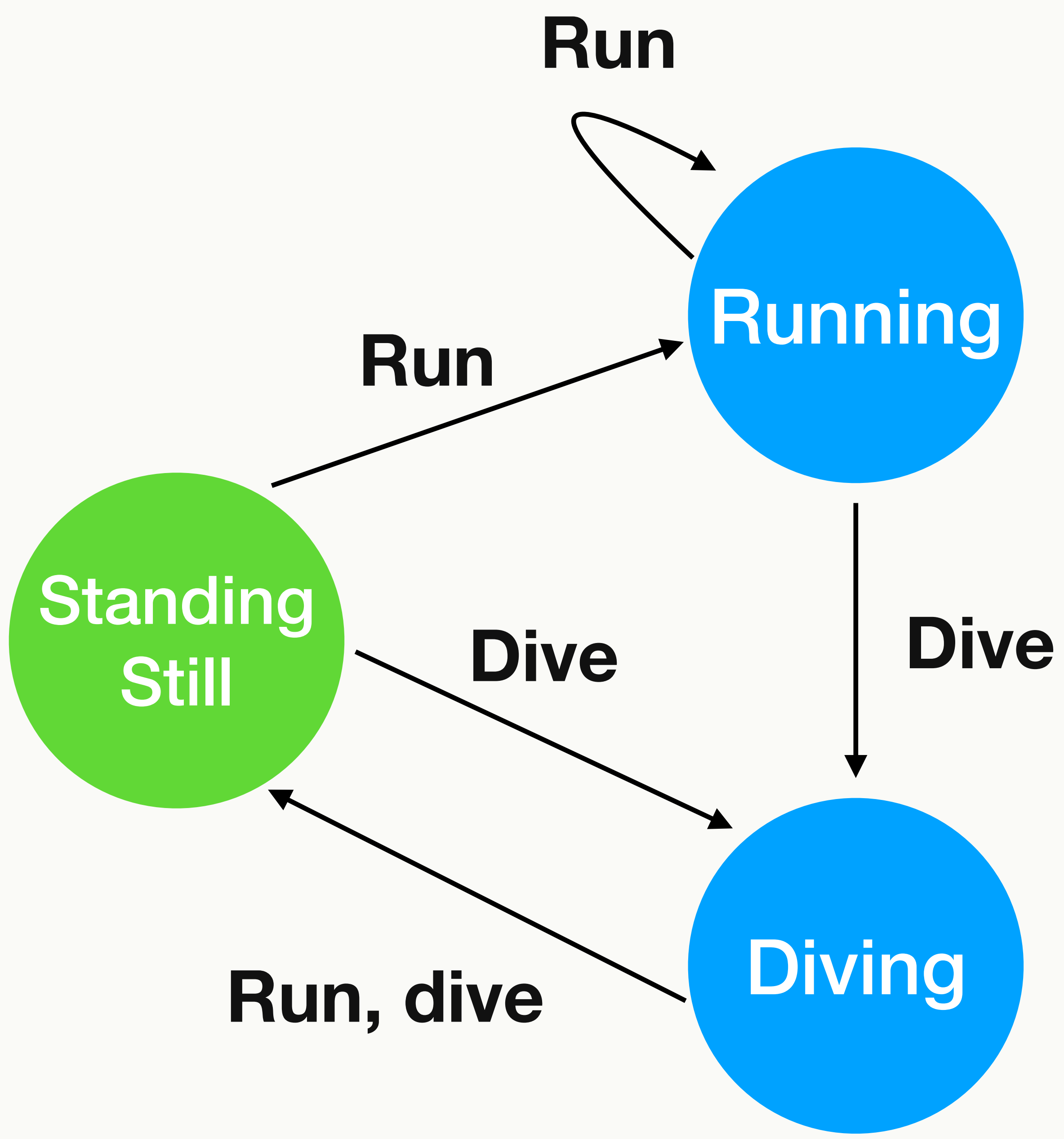
Dive forever $\rightarrow \{1.5,0,1.5,0,\ldots\}$ Avg. reward = 0.75

Run, dive, x $\rightarrow \{1,1.5,0,1,1.5,0,\ldots\}$ Avg. reward = 5/6

Note: average reward slightly different than discounted reward and often has cleaner math

- Roughly equivalent to discount factor of 1

Which locomotion policy is best (expected discounted return)?

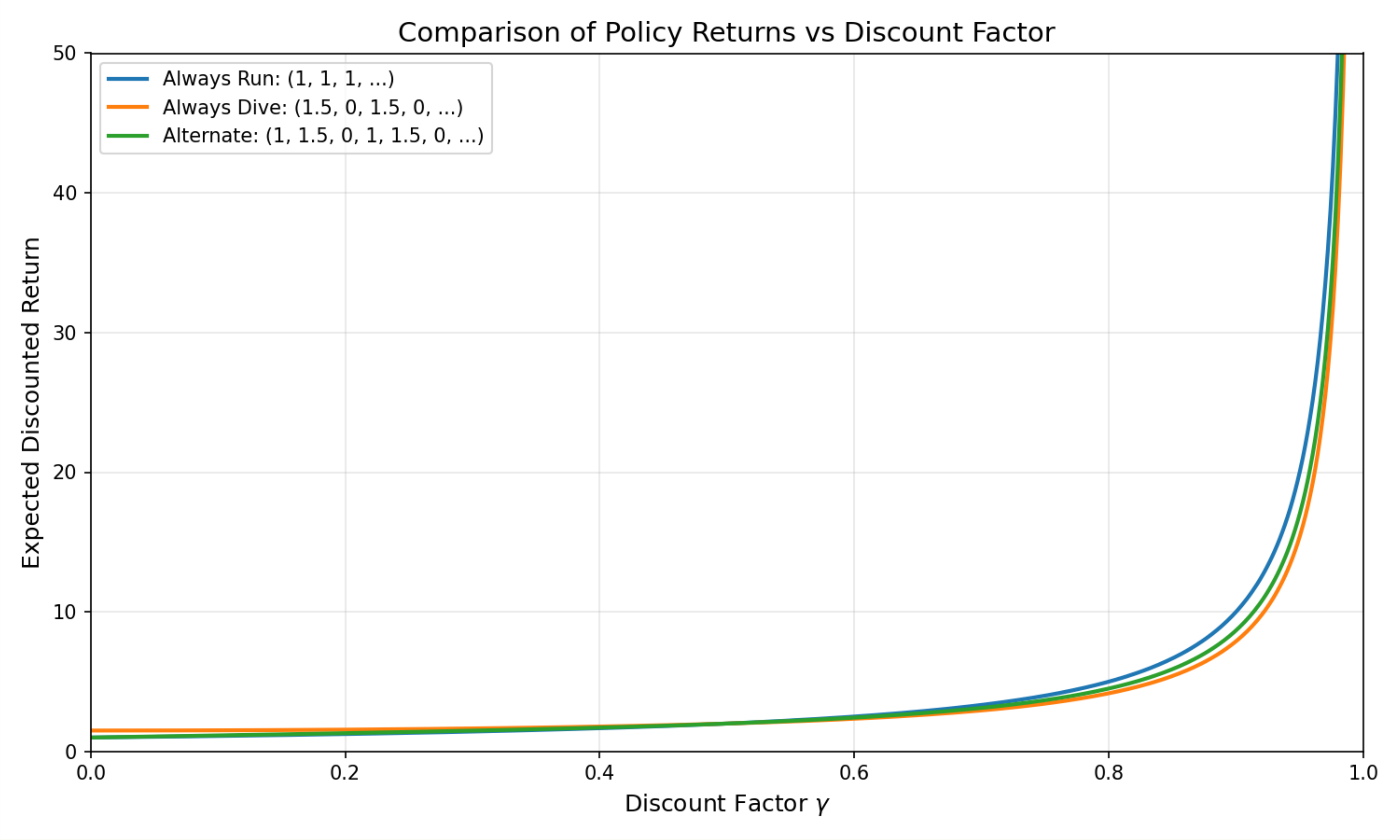


Recall: lower discount factor → we are more impatient

- Discount factor γ → indifferent between 1 reward now and $1/\gamma^T$ reward in T steps

What do you think will happen as we lower the discount factor?

Lowering the discount factor



At $\gamma = 0.5$, indifferent between all three policies

- Math: geometric series

Discount < 0.5: policy order reverses with dive > run_dive > run

Most common discount rates in practice:

- 1 for LLMs
- 0.99 for robotics

Questions?

Solving larger MDPs

Enumerating all of the policies is a great way to solve small MDPs, but there are exponentially many in the worse case (curse of dimensionality)

Key idea: define an operation that will improve the policy
- Ideally, operator should never get stuck in local minimum
- How are we going to do this and prove it?

Value function

The **value function** (or *state-value function*): $V_\pi : S \to \mathbb{R}$

- Intuition: how good is state s under policy π?

Define $V_\pi(s) = \mathbb{E}_{s_0=s,a_0,r_1,\ldots}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right]$

The value function is just the expected return of a policy starting from *any* state:
$J(\pi) = \mathbb{E}_{s_0}[V_\pi(s_0)]$, where s_0 is a sampled start state

Note: expected return may not exist (e.g., if it's infinite)—let's ignore this

I claim the value function is unique, but will not prove it, yet 🙃

Properties of the value function

$$V_\pi(s) = \mathbb{E}_{s_0=s,a_0,r_1,\ldots}\left[\sum_{t=0}^{\infty}\gamma^t r_{t+1}\right] = \sum_a \pi(a\,|\,s)\sum_{s'} P(s'\,|\,s,a)\Big[\mathbb{E}[R_a(s,s')] + \gamma V_\pi(s')\Big]$$

We can **unroll** the value function, computing the expectation by summing over the expected returns for each action
- Action expected return = reward for this step + discounted expected future reward
= reward for this step + discounted value function

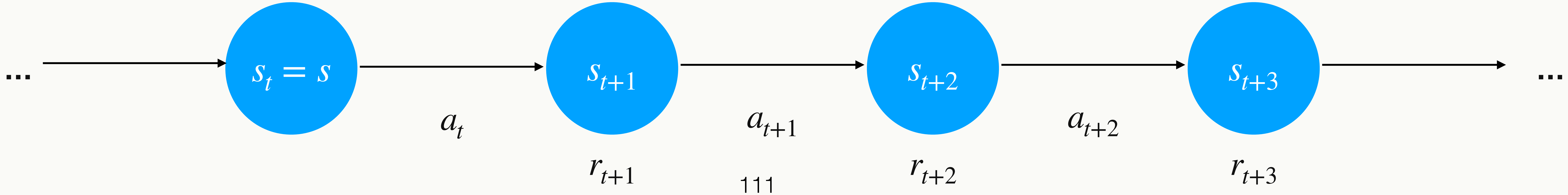
Why can we do this? The expected return after entering a state does not depend on how we got there (Markov property)

Calculating the value function: MC

Suppose we have a set of **trajectories** $\tau = (s_0, a_0, r_1, \ldots)$ from a policy π

Idea #1: **Monte Carlo (MC)**
- Since the value function is an expectation, just average the expected return over all trajectories
- For $\hat{V}_\pi(s)$, "observation" is discounted return from s, i.e., each time we see s in $\tau = (\ldots, s_t = s, a_t, r_{t+1}, \ldots)$, we calculate $G(s_t) = \sum_{t=t}^{\infty} \gamma^t r_{t+1}$, and then average these

- Pros: requires no knowledge of MDP or policy, unbiased
- Cons: need to wait until the end of the episode to have estimated return, high variance



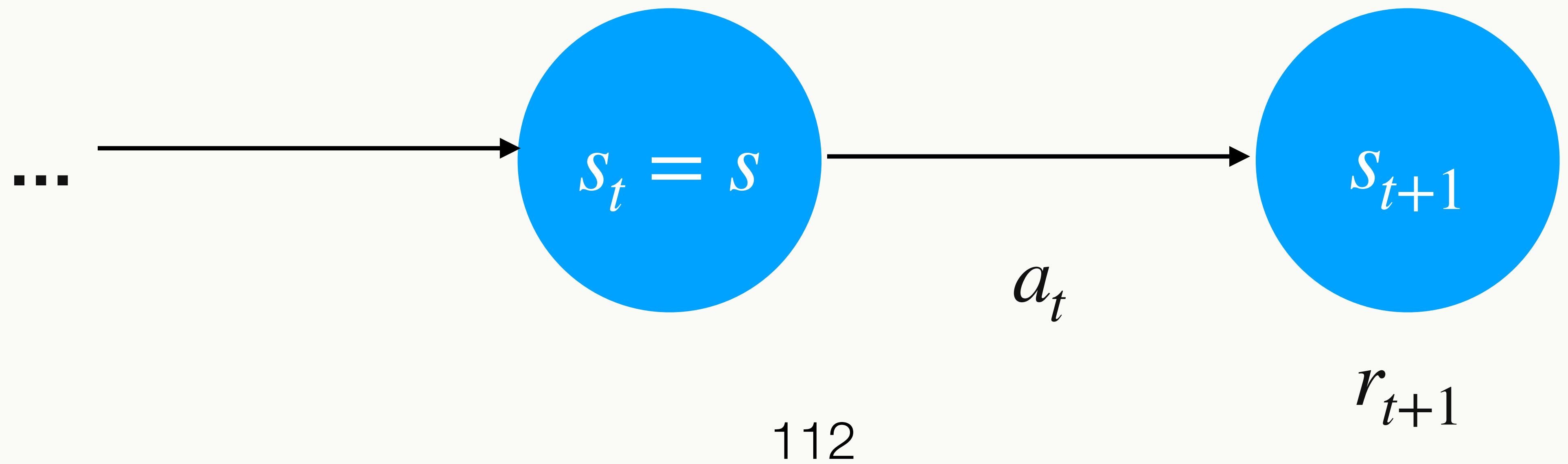
Calculating the value function: TD

Idea #2: **temporal difference (TD)**: *bootstrap* value function estimation, i.e., instead of waiting for the episode to end, use the *current estimate of the value function* to simulate the rest of the episode

For $\hat{V}_\pi(s)$, each time we see s in $\tau = (\ldots, s_t = s, a_t, r_{t+1}, \ldots)$, we calculate $r_{t+1} + \gamma \hat{V}(s_{t+1})$ and use that as our observation

- Idea: unrolling on a sample

- Pros: requires no knowledge of MDP or policy, do not need to wait until the end of the episode, less variance
- Cons: highly biased by initialization of $\hat{V}$



Calculating the value function: direct

If we know the MDP, we can use them to calculate the value function directly, without samples

We know the value function satisfies:

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a) \Big[\mathbb{E}[R_a(s, s')] + \gamma V_\pi(s') \Big]$$

- This is just a system of linear equations, can use any suitable method for solving it

Pros: extremely fast, no estimation error
Cons: need to know entire MDP and policy

The optimal value function

For the optimal policy, I claim its value function $V_{\pi*}$ must satisfy:

$$V_{\pi*}(s) = \max_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\Big[\mathbb{E}[R_a(s, s')] + \gamma V_{\pi*}(s')\Big]$$ (the Bellman equation)

Why?

- By definition of a value function $V_{\pi*}(s)$ must equal the expected discounted return from s then following $\pi*$
- But it doesn't satisfy the Bellman equation, so it must not pick the $\arg\max$ action
- We can get a better expected discounted return by picking the $\arg\max$ action
- Contradiction: optimal policy must be optimal

Optimal value function → optimal policy

Suppose we have a function that satisfies:

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a) \left[\mathbb{E}[R_a(s, s')] + \gamma V_\pi(s') \right]$$

How can we "read" an optimal policy from it?

The max has to be "active", i.e., there is some action (or actions) that achieves the max

Any of these actions are optimal

Proving stuff about value functions

Mathematical power tools: fixed-point theory (1910s-20s)
- origins: topology
- Theorems are really easy to apply and useful
- Required new math to be invented, most of the proofs are really hard (algebraic topology, homology, degree theory)

Key contributors: L. E. J. Brouwer, Stefan Banach, Shizuo Kakutani
- Some of the most influential mathematicians of the 20th century

Basis of:
- modern microeconomics (shows that markets have equilibria)
- optimization theory (KKT conditions, gradient descent convergence rate)
- RL (value function existence, finding optimal policies)

Brouwer's fixed point theorem

If f is continuous and $f: X \rightarrow X$, where X is convex and compact (closed and bounded), **a fixed point exists**
- Non-constructive—don't know how to find the fixed point
- May not be unique

Let's apply this to value functions. Let $X \in \mathbb{R}^{|S|}$ is the set of all possible value functions. How do we make this compact (closed and bounded?)

If our rewards are bounded, the value functions are too
- Largest value function could be is if it got that highest reward every step
- Smallest value function could be is if it got the lowest reward every step
- So just define a box (convex) with suitable upper and lower bounds → compact

Brouwer's: value function existence

If f is continuous and $f: X \to X$, where X is convex and compact (closed and bounded), **a fixed point exists**

Let X be the space of value functions (which we proved is convex and compact)

Let f be value function unrolling:

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[\mathbb{E}[R_a(s, s')] + \gamma V_\pi(s')\right]$$

- This is continuous in $V_\pi(s)$: just addition and multiplication

Therefore, value function must exist

Brouwer's: optimal value function

If f is continuous and $f : X \to X$, where X is convex and compact (closed and bounded), **a fixed point exists**

Let X be the space of value functions (which we proved is convex and compact)

Let f be Bellman equation:

$$V_{\pi^*}(s) = \max_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a) \left[\mathbb{E}[R_a(s, s')] + \gamma V_{\pi^*}(s') \right]$$

- This is continuous in $V_{\pi^*}(s)$: just addition, multiplication and max

Therefore, optimal value function must exist

Banach's fixed point theorem

Key idea: in well-behaved fixed point operators (f is a contraction), we can find a fixed point by repeatedly applying f and the resulting fixed point is unique

A mapping $f : X \to X$ is called a contraction if, for all $x \in X, y \in X$, $d(f(x), f(y)) \leq qd(x, y)$, where d is a distance metric and $q \in [0,1)$

Intuition: no matter where you start, you're headed to the same place

Banach’s: value function

Both unrolling $V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[\mathbb{E}[R_a(s, s')] + \gamma V_\pi(s')\right]$ and the Bellman equation $V_{\pi*}(s) = \max_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a)\left[\mathbb{E}[R_a(s, s')] + \gamma V_{\pi*}(s')\right]$ are contractions

- Proofs are short and easy

Thus, value function and optimal value function are unique 🤷‍♂️

Corollaries

1. Value functions and optimal value functions always exist and are unique

2. There can be multiple optimal policies if there are ties in the optimal value function (multiple “optimal actions”)

3. There may not be a stochastic optimal policy—stochastic optimal policies can only have put probability mass over “optimal actions”

4. The Bellman equation has some additional properties that, when combined with fixed point theory, yield a linear programming formulation

5. Highly efficient algorithms for known MDPs exist