Advanced Topics in Reinforcement Learning

Lecture 5: Dynamic Programming II

Josiah Hanna
University of Wisconsin — Madison

Announcements

- Sign-up for a presentation:https://docs.google.com/spreadsheets/d/1PMI8XO9lP84GW5jYFJi1qPo6E19ZKacw5nRKxY7YTu8/edit?gid=0#gid=0
  - Many of the latter slots have been taken

Learning Outcomes

After today's class, you will be able to:

1. Be able to explain the steps of policy evaluation, policy iteration, and value iteration.

2. Be able to explain the explain the policy improvement theorem as a basis for policy updates in RL.

Marty’s Presentation

- Reward is Enough. Silver, Singh, Precup, and Sutton. 2021.
- Link to slides.

Quick Review

- Define the following:
  - $v_\pi(s)$
  - $q_\pi(s, a)$
  - $q_\star(s, a)$
- What is the relationship between $\pi^\star$ and $q_\star$?

Dynamic Programming in RL

- Use value functions to find improved policies.
- Turn Bellman equations into value function updates.
- Bellman equation for policy value becomes policy evaluation:

$$v_{k+1}(s) \leftarrow \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma v_k(s')]$$

- Bellman optimality equation becomes value iteration:

$$v_{k+1}(s) \leftarrow \max_a \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma v_k(s')]$$

Policy Evaluation (Prediction)

- Given a policy, compute its state- or action-value function.

$$v_{k+1}(s) \leftarrow \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma v_k(s')]$$

$$q_{k+1}(s, a) \leftarrow \sum_{s'} \sum_r p(s', r \mid s, a)[r + \gamma \sum_{a'} q_k(s', a')]$$

- When to stop making updates?
- Do these updates converge?
    - Yes, updates are a **contraction mapping** with respective fixed points v_π, q_π.
    - Convergence proof for value-iteration. Can you generalize it?

Policy Iteration

- We have $v_\pi(s)$ for the current policy π. How can we improve π?
- Alternate until π stops changing:
  - Run policy evaluation updates to find v_π.
  - Set $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_\pi(s')]$
  - Why does this work?

Policy Improvement Theorem

- Suppose for π that $\exists s, a$ such that $q_\pi(s, a) \geq v_\pi(s)$.
- Let $\pi'(s) = a$ and $\pi'(\tilde{s}) = \pi(\tilde{s})$ for all other states $\tilde{s}$.
- What is true about π'? Why?
- If π is sub-optimal, does there exist s, a such that $q_\pi(s, a) \geq v_\pi(s)$?
  - Yes, this follows from Bellman Optimality. Must be at least one state where π is not greedy w.r.t. its action-value function.
  - Optimal value function: $v_\star(s) = \max_a q_\star(s, a) \forall s$

Policy Improvement Theorem

$$
\begin{aligned}
v_\pi(s) &\le q_\pi(s, \pi'(s)) \\
&= \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = \pi'(s)] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\
&\le \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \mid S_t = s] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_{t+1}, A_{t+1} = \pi'(S_{t+1})] \mid S_t = s] \\
&= \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \bigm| S_t = s\big] \\
&\le \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(S_{t+3}) \bigm| S_t = s\big] \\
&\vdots \\
&\le \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \bigm| S_t = s\big] \\
&= v_{\pi'}(s).
\end{aligned}
$$

Value Iteration

- What's wrong with policy iteration?
  - Evalution convergence is wasteful! — we just need to know the maximizing action.
- Value iteration combines policy evaluation and improvement into one step.

$$v_{k+1}(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_k(s')]$$

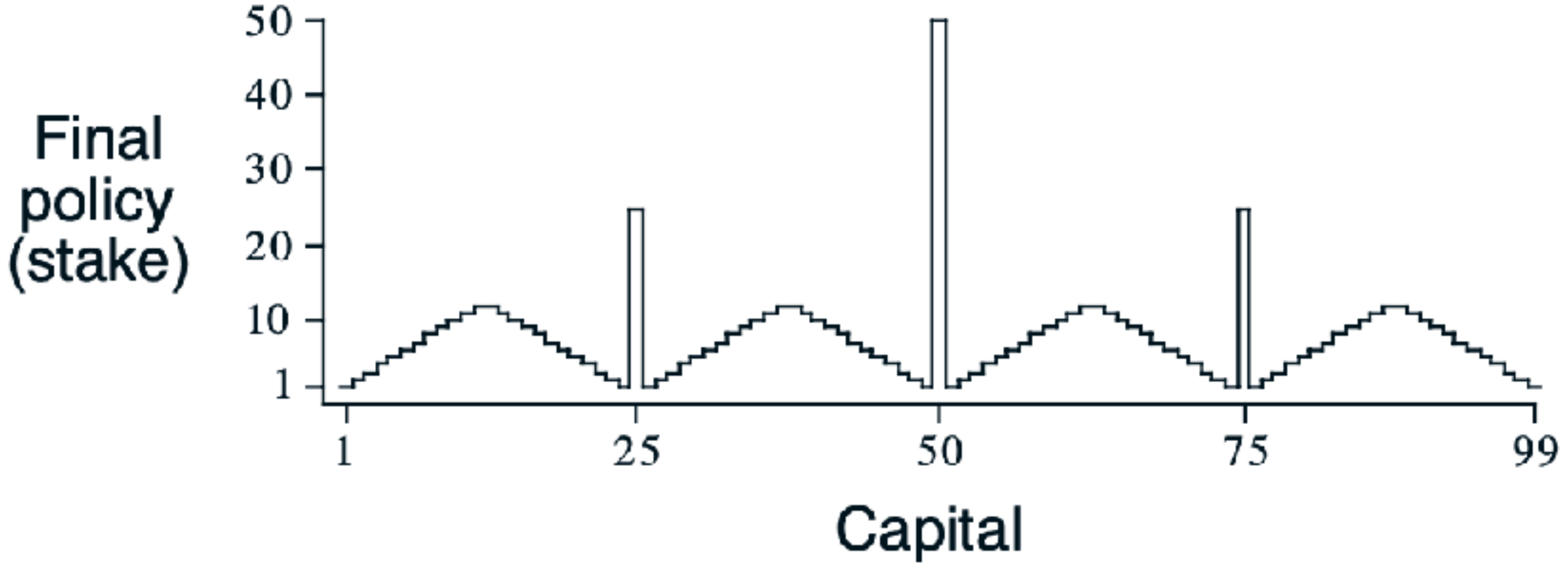
- In-place vs out-of-place updates?
  - In-place propagates value changes faster.
  - Out-of-place is easier to analyze.

Policy Iteration Demo

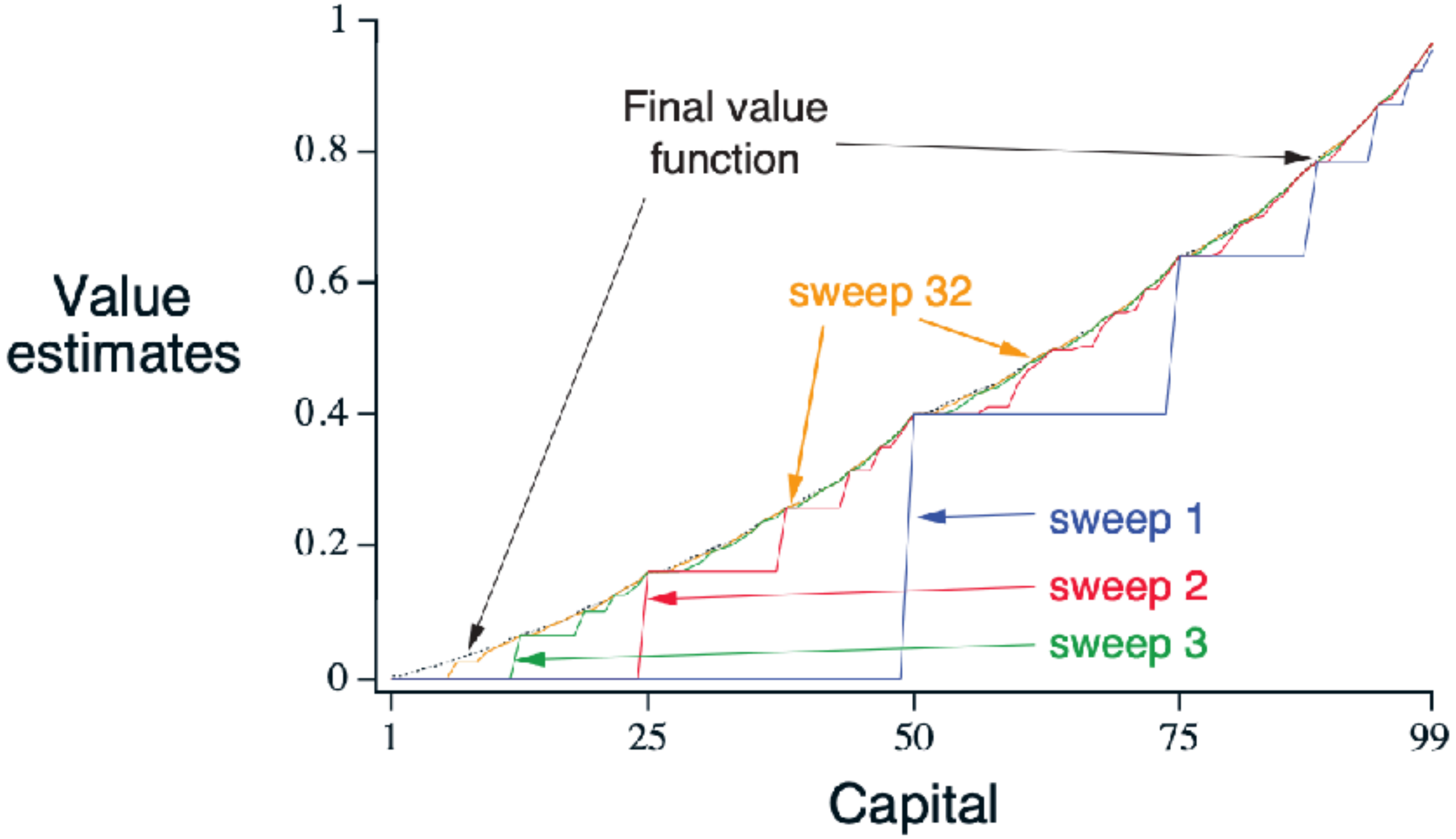
https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_dp.html

Gambler's Problem

$$v_{k+1}(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_k(s')]$$



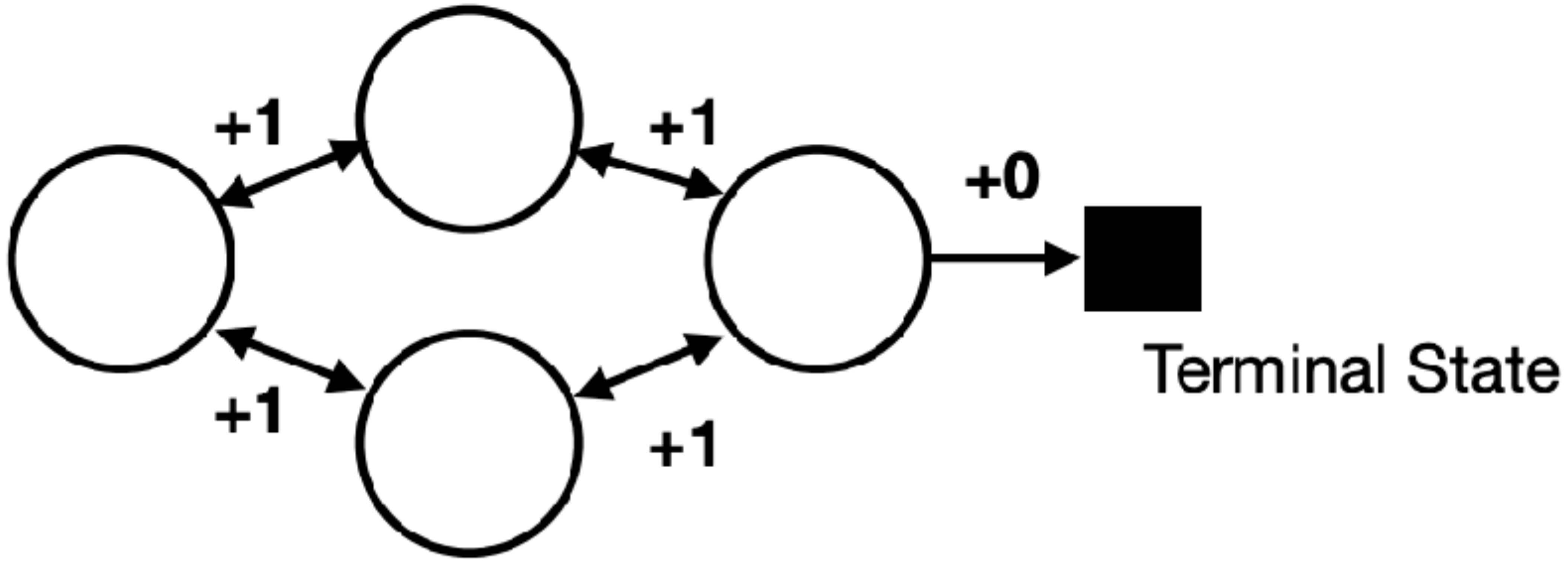
Why is this an optimal policy?

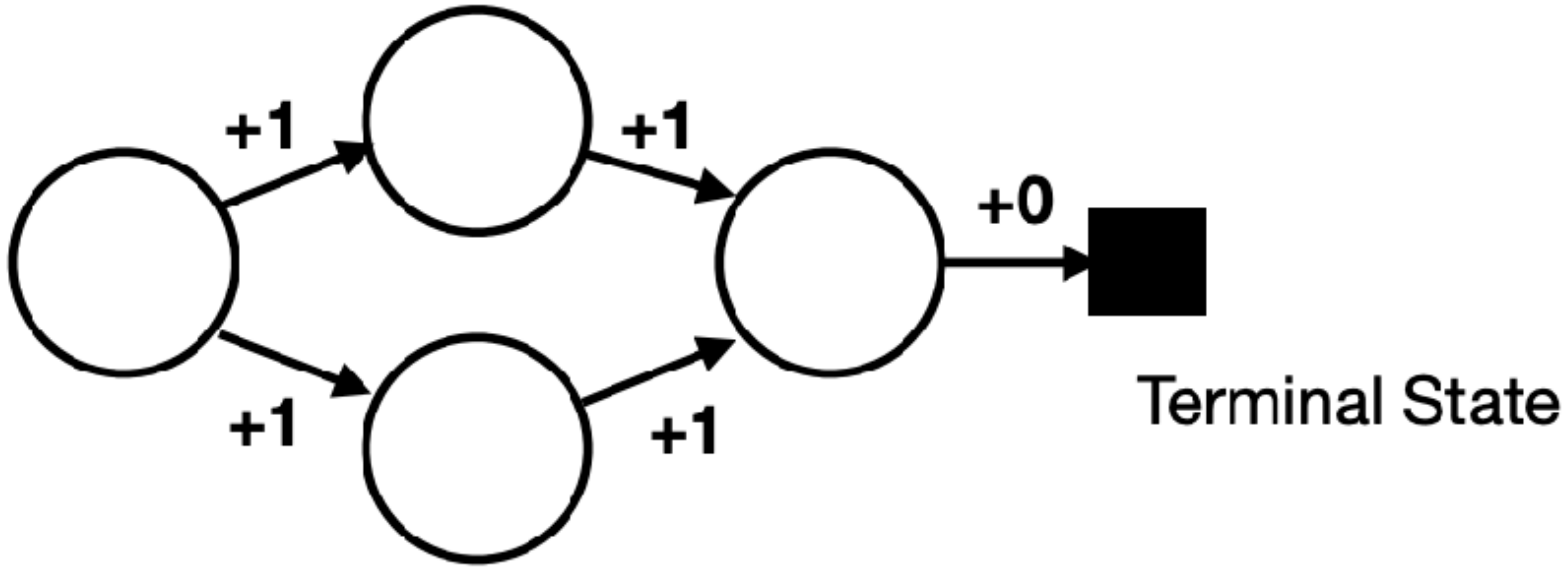


Why does the VF look like this after each sweep?

Asynchronous DP

- Basic DP methods require a sweep over the entire state space *per iteration*.
    - Infeasible with a huge state space.
- Actually unnecessary — can update states in any order and still converge as long as all states are updated infinitely often in the limit.
- Why does this help?





Generalized Policy Iteration

- What is it?
  - Can be permissive in how we mix evaluation and improvement.
  - As long as v_k becomes closer to v_π and π becomes greedy w.r.t. v_k then we will converge to $v_\star$ and $\pi^\star$.
- Essentially all RL algorithms are instances of this framework.
- How do you think function approximation will affect GPI?

Summary

- Learning value functions allow us to compute optimal policies.
- Policy Evaluation: find value function for a fixed policy.
- Policy Iteration: compute optimal policy by iterating 1) policy evaluation and 2) greedy policy improvement.
- Value Iteration: directly learn optimal value function.
- Dynamic programming methods don't solve the full RL problem but they are the basis for most of the methods we will see in this class.

Action Items

- Read Chapter 5 of course textbook.
- Send a reading response by 12pm on Monday.
- Sign-up for a presentation: https://docs.google.com/spreadsheets/d/1PMI8XO9lP84GW5jYFJi1qPo6E19ZKacw5nRKxY7YTu8/edit?usp=sharing
- Begin thinking about final project; proposal due October 2nd.
    - Use Piazza to look for a partner.