Last Module Recap (“Bandits”)

- Bandits:
  - Epsilon-greedy
  - Regret
  - UCB
  - Thompson Sampling
- Contextual Bandits:
  - LinUCB

Microsoft

The Reinforcement Learning Problem

ROLAND FERNANDEZ

Researcher, MSR AI

Instructor, AI School

Fundamental Challenges

- Representation
- Generalization
- Exploration
- *Temporal Credit Assignment*

Outline

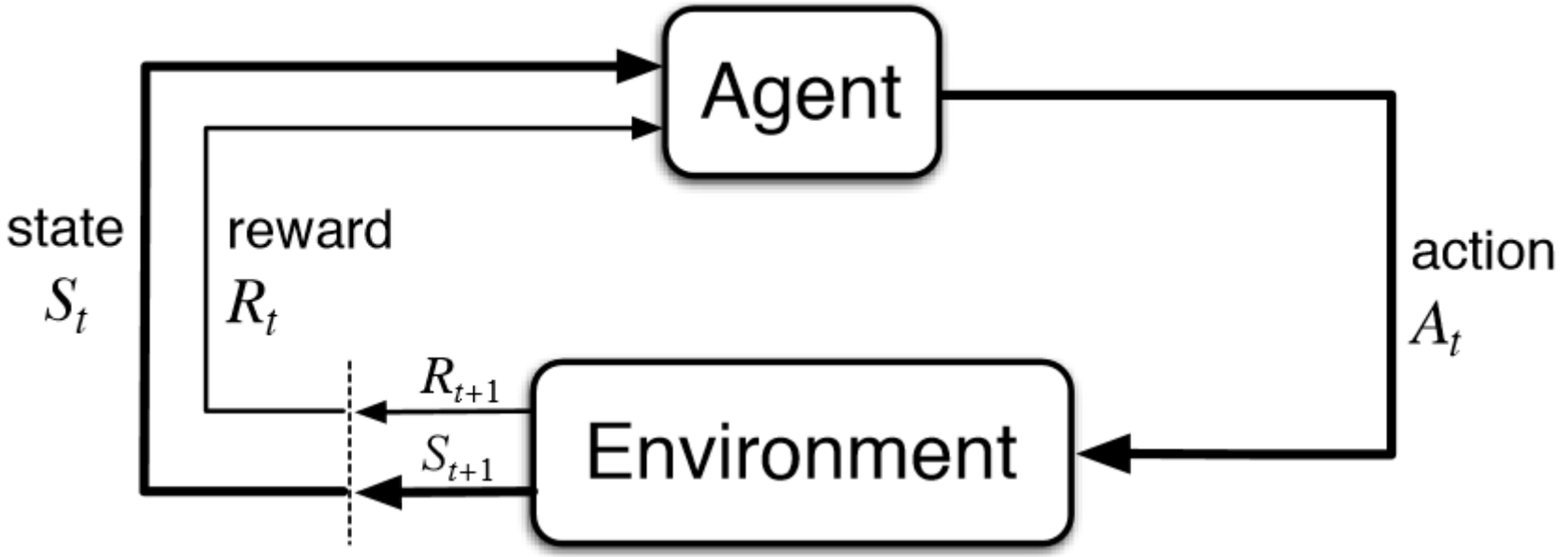
- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

The Agent-Environment Interface

- Notation:
  - Time step t
  - State $S_t \in \mathcal{S}$
  - Action $A_t \in \mathcal{A}(S_t)$
  - Reward $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$
  - Policy π



- General interface for all RL problems
- Used by Open AI Gym

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Environment / Agent Separation

- RL problem design
  - Set of: states, rewards, actions + state-transition probabilities (model)
- Common separation:
  - Agent is the decision making entity
  - Environment is everything else
- Robot:
  - Body is part of Environment
  - Enables Agent to learn to sense and operate the body
  - Enables RL problem where Robot's body defines goals/rewards

RL Problem: Broad Scope

- State:
  - low level sensors, symbolic descriptions of objects, memory of events
- Actions:
  - Motor controls, UI design choices, buy/sell stock

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Goals and Rewards

- Rewards:
  - Defined by Task designer to abstract the goal
  - A real number delivered on each timestep by the Environment
  - Should reflect what **has** been achieved (not how)
- Reward Hypothesis:
  - Goals & purposes can be well represented thru the maximization of a cumulative sum of the reward signal
- Reward signal has proven to be effective and flexible

Returns

- **Return** is the long-term accumulation of **reward** starting from time t

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + ... + R_T$$

  - A simple return function could be the sum of the rewards
- What about continuing tasks?

Returns

- Introduce *discounted return*, using a discount factor, γ ([0,1]):

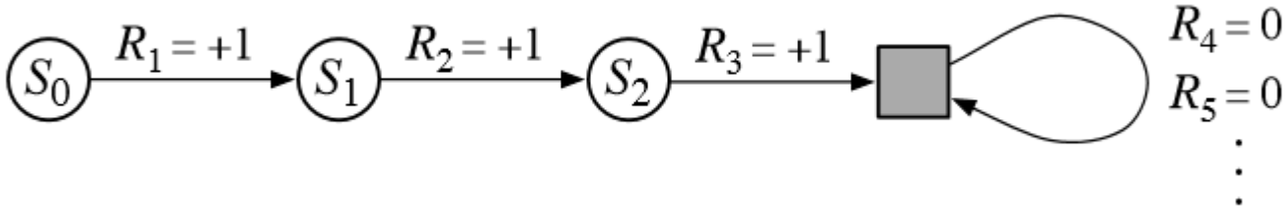
$$
\begin{aligned}
G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \\
&= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}
\end{aligned}
$$

- Key concept for RL: recursive relationship

$$
\begin{aligned}
G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \\
&= R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \ldots) \\
&= R_{t+1} + \gamma G_{t+1}
\end{aligned}
$$

Unified Notation

- How to unify episodic and continuing tasks?
- Absorbing state for episodic:



- Unified notation:

$$G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k, \qquad T = \infty \text{ or } \gamma = 1$$ (but not both)

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

The Markov Property

- State:
  - Some set of values that define the current situation
  - Provide basis for agent selecting action from its policy
  - Immediate or processed sensations
- If state representation is as effective as having a full history, it is said to have the Markov Property
- Examples:
  - Tic-tac-toe
  - Space Invaders

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

The Markov Decision Process (MDP)

- Markov Decision Process
  - A RL problem that satisfies the Markov property
- Finite MDP:
  - Finite members: $\{\mathcal{S}, \mathcal{A}, \mathcal{R}\}$
  - Dynamics defined by state-transition probabilities:

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

    - From these probabilities, we can compute anything about the MDP we might need

Example: Recycling Robot

- **Goal**: collecting empty soda cans in cafeteria
- Has on-board recycling bin
- Actions:
  - Go to home base to recharge battery
  - Search for cans
  - Wait for someone to bring it a can

Example: Recycling Robot

- MDP Task:
  - States: {low, high}
  - Actions:
    - A(low) = {search, wait, recharge}
    - A(high) = {search, wait}
  - Rewards: {-3, 0, 1}

r_{wait} Expected # of cans while waiting
r_{search} Expected # of cans while searching
α Probability of high -> high on search
β Probability of search cycle on low battery

Example: Recycling Robot

Transition Probabilities & Expected Rewards (Task Model)

s	a	s'	$p(s'\|s,a)$	$r(s,a,s')$
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	$r_{\texttt{wait}}$
low	wait	high	0	$r_{\texttt{wait}}$
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	0.

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Example: Recycling Robot

Transition Graph

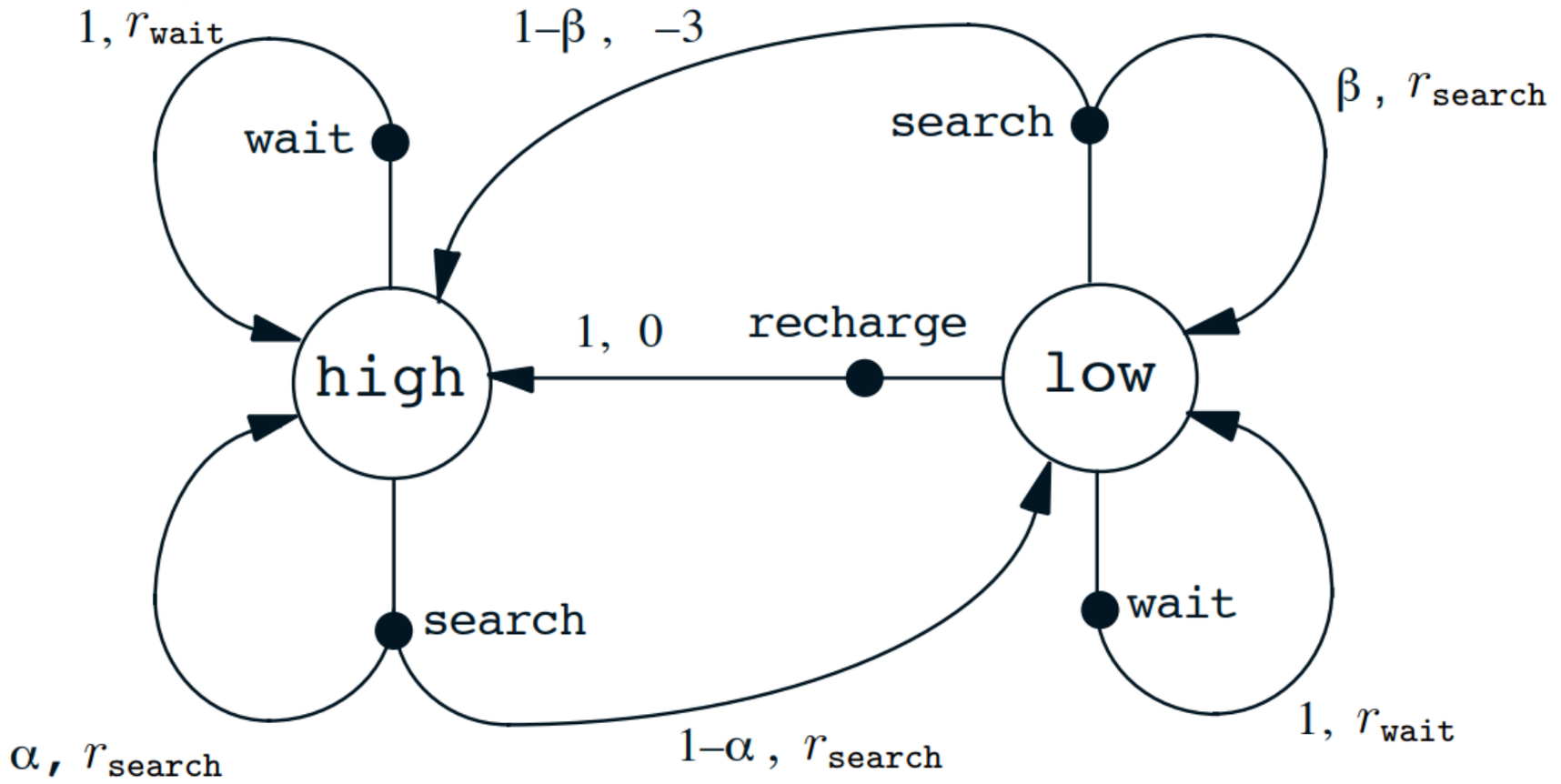


Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Value Functions

- Value Functions:
  - One of the most fundamental concepts in RL
  - Functions of state (or state-action pairs)
  - Estimate how good it is to be in a state, or take some specific action in a state (expected ***return***)
- Policies:
  - A mapping from state to probability of selecting each available action:

    $$\pi(a|s)$$

  - RL methods specify how agent's policy changes with experience

Value Functions

- State-value function for policy π:

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

- Action-value function for policy π:

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

- Value functions can be ***learned*** from experience

Value Functions (important)

- Satisfy recursive relationships similar to G_t

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r \,|\, s, a) \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s'] \Big] \\
&= \sum_a \pi(a|s) \sum_{s', r} p(s', r \,|\, s, a) \Big[r + \gamma v_\pi(s') \Big], \quad \text{for all } s \in \mathcal{S},
\end{aligned}
$$

→ Bellman Equation for v_π

Bellman Equations

- Express the value of a state in terms of the value of successor states
- Defined for state-value function and action-value function
- Why important?
  - Form the basis for ***calculating, approximating, and learning*** v_π

Backup Diagram

- Backup diagram for v_π
- Relationships between states, actions, next states, …
- Used to graphically illustrate many RL algorithms

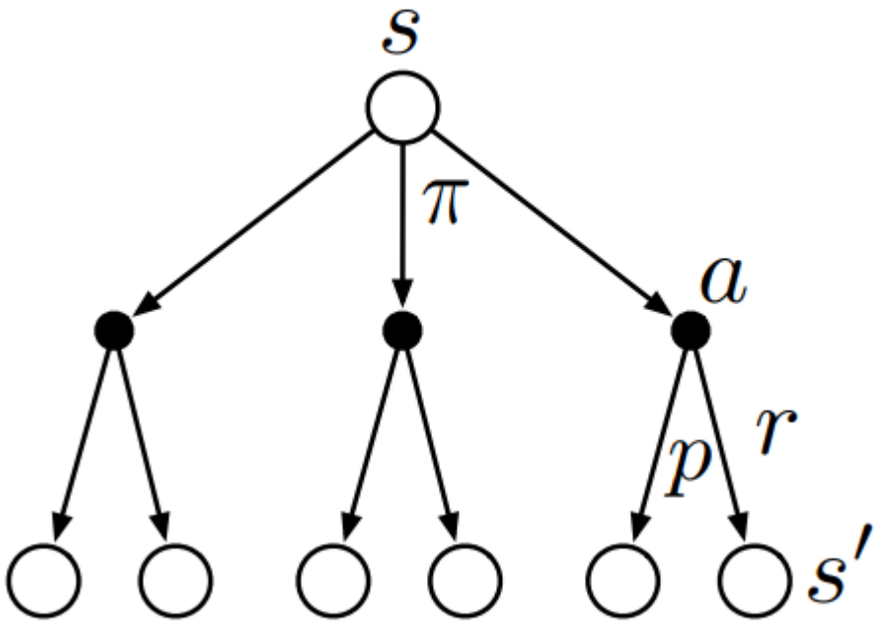
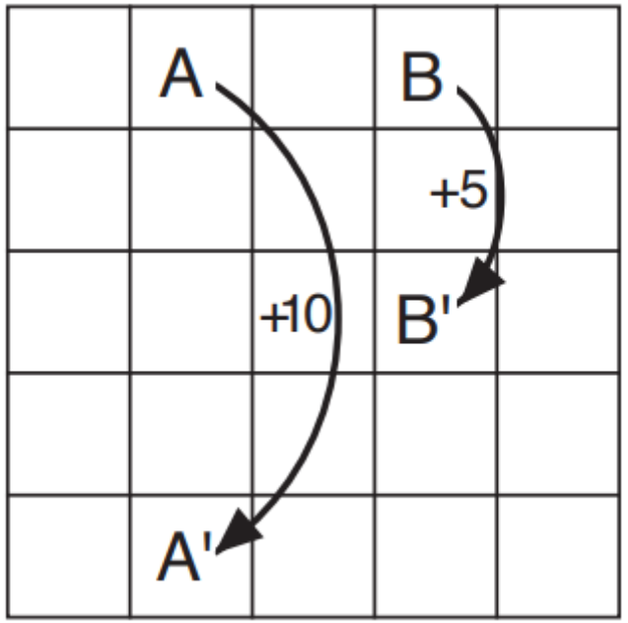
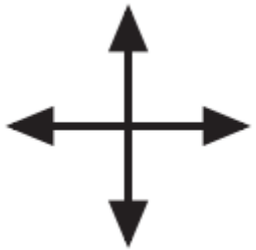


Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Gridworld Example



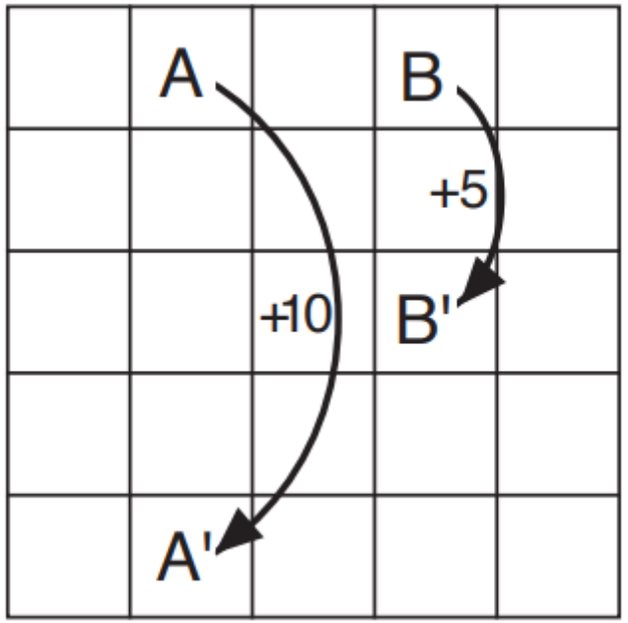


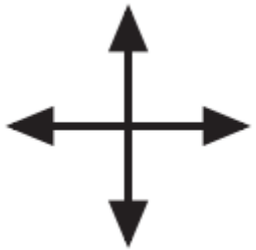
Actions
π

- Rewards:
  - -1 if try to move off edges
  - +10 from A
  - +5 from B
  - 0 otherwise
  - $\gamma = .9$
- Continuing task

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Gridworld Example





Actions
π

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

Value function of each state under policy π

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Optimal Value Functions

- What does it mean to solve an RL task?
  - Finding a policy that achieves a lot of reward in the long term
- Value functions define a partial ordering over policies
  - Policy π is defined to be better or equal to π′ if $v_\pi(s) \geq v_{\pi'}(s)$ for all s

Optimal Value Functions

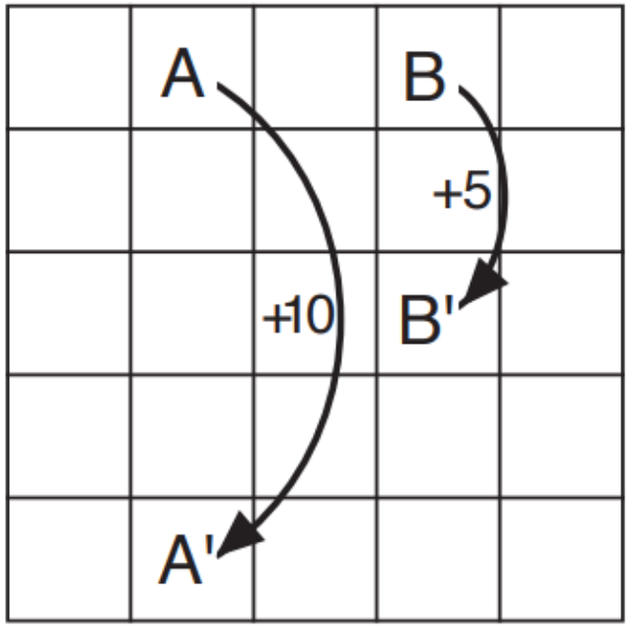
- For any task, there is a least one policy that is >= all other policies:
  - This policy is an ***optimal policy*** for the task, denoted by π_*
  - All optimal polices share the same ***optimal state-value function***:

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

  - All optimal polices share the same ***optimal action-value function***:

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

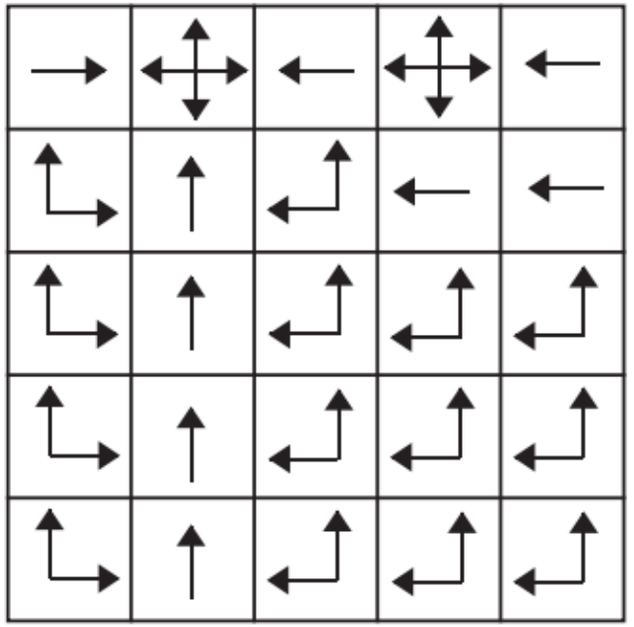
Solving Gridworld



Gridworld

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

v_*



π_*

- Compute optimal value function using bellman equation
- Choose greedy policy

Image Credit: Sutton and Barto, Reinforcement Learning, An Introduction 2017

Solving RL Problems

- Can we use this approach on all RL Problems?
  - Usually not – it requires:
    - Accurate knowledge of Environment dynamics (model)
    - Sufficient computation for something like an exhaustive search
    - Task have the markov property
  - We explore a variety of methods for approximating the bellman equations in subsequent modules

Outline

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Optimality and Approximation

- Optimality
  - Rarely achieved for real world RL tasks (model, computation/memory, markov)
  - Has role in organizing and understand of learning algorithms
- Approximation
  - Often works quite well for real world tasks
  - On-line nature of RL:
    - More learning for frequently encountered states
    - Results in better performance on those states

Summary

- The Agent-Environment Interface
- Goals, Rewards, Returns
- The Markov Property
- The Markov Decision Process
- Value Functions
- Optimal Value Functions
- Optimality and Approximation

Further Reading

- Chapter 3 of “Reinforcement Learning: An Introduction” (2018)