Warm-up as You Walk In

Given

- Set `actions` (persistent/static)
- Set `states` (persistent/static)
- Function `T(s,a,s_prime)`

Write the pseudo code for:

- `function V(s) return value`

that implements:

$$V(s) = \max_{a \in actions} \sum_{s' \in states} T(s,a,s')V(s')$$

Announcements

Assignments:

- P3: Optimization; Due 10/17 Thu, 10 pm
- HW6 (online) 10/15 Tue, 10 pm
- HW7 (online) 10/22 Tue, 10 pm; Will be released today

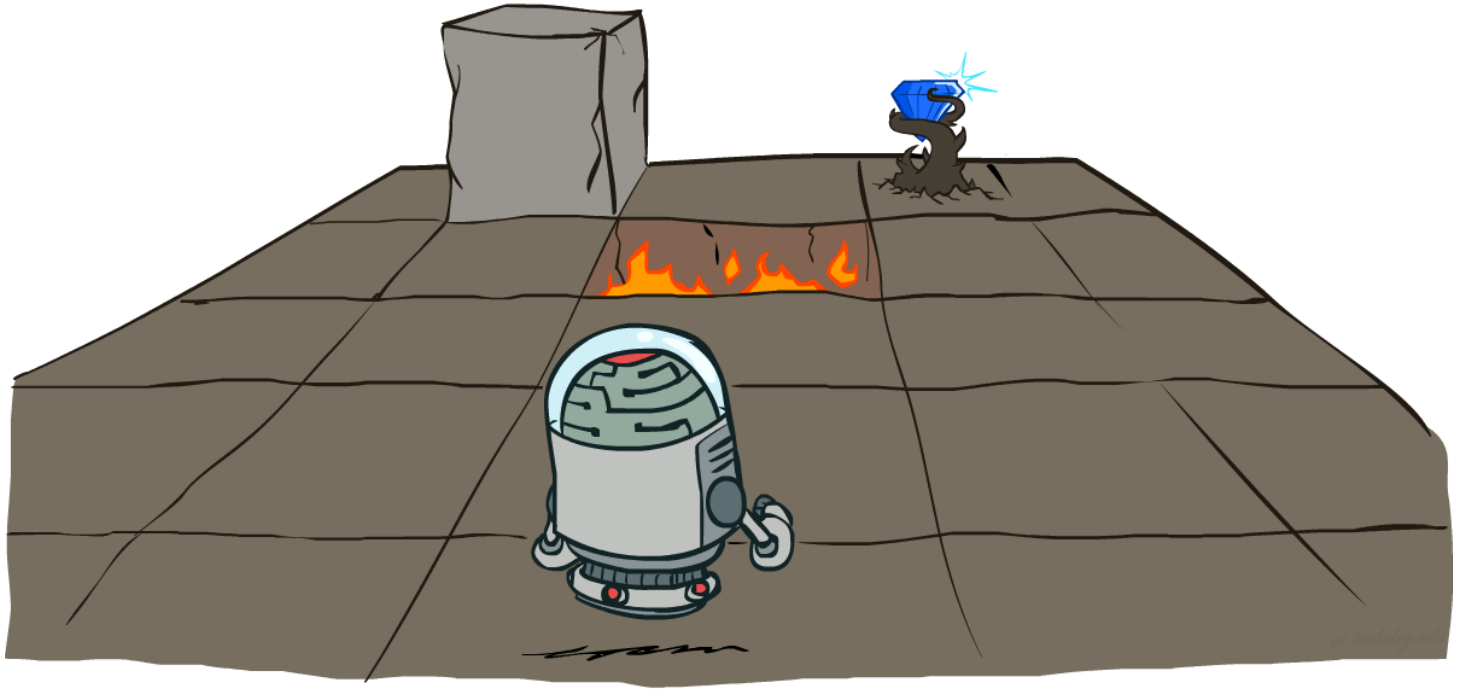
Lectures: 4 lectures by Dr. Fei Fang on MDP/RL, followed by 4 lectures by Dr. Pat Virtue on Bayes' Nets

Recitations canceled on October 18 (Mid-Semester Break) and October 25 (Day for Community Engagement). Recitation worksheet available (reference for midterm/final)

New: Piazza post for In-class Questions

AI: Representation and Problem Solving

Markov Decision Processes



Instructors: Fei Fang & Pat Virtue

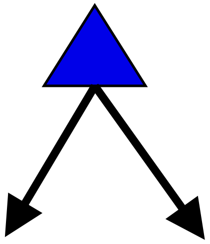
Slide credits: CMU AI and http://ai.berkeley.edu

Learning Objectives

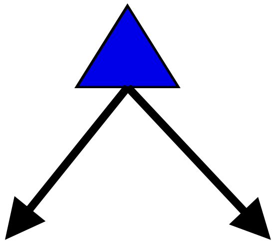
- Describe the definition of Markov Decision Process
- Compute utility of a reward sequence given discount factor
- Define policy and optimal policy of an MDP
- Define state-value and (true) state value of an MDP
- Define Q-value and (true) Q value of an MDP
- Derive optimal policy from (true) state value or (true) Q-values

Next Lecture

Recall: Minimax Notation



$$V(s) = \max_{a} V(s'),$$
where $s' = result(s, a)$

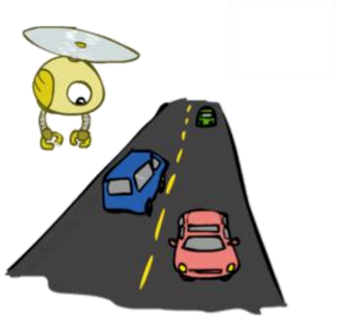


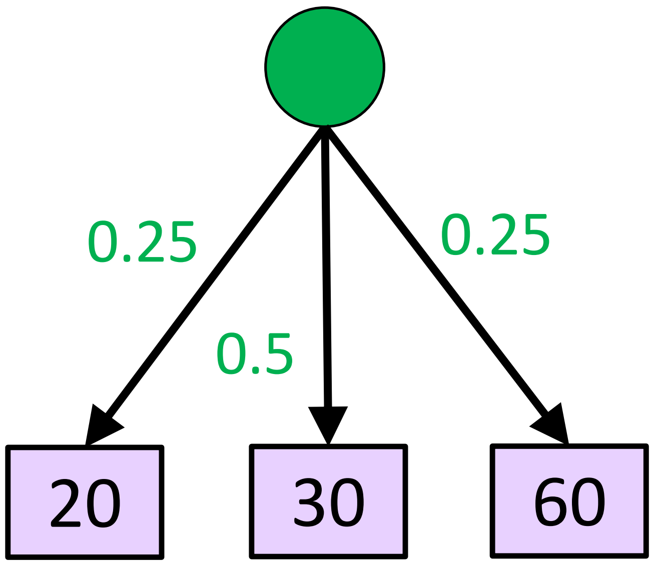
$$\hat{a} = \underset{a}{\operatorname{argmax}}\, V(s'),$$
where $s' = result(s, a)$

Recall: Expectations

Time: 20 min x + 30 min x + 60 min x

Probability: 0.25 0.50 0.25





Max node notation

$$V(s) = \max_a V(s'),$$

where $s' = result(s, a)$

Chance node notation

$$V(s) = \sum_{s'} P(s')\, V(s')$$

Piazza Poll 1

Expectimax tree search:

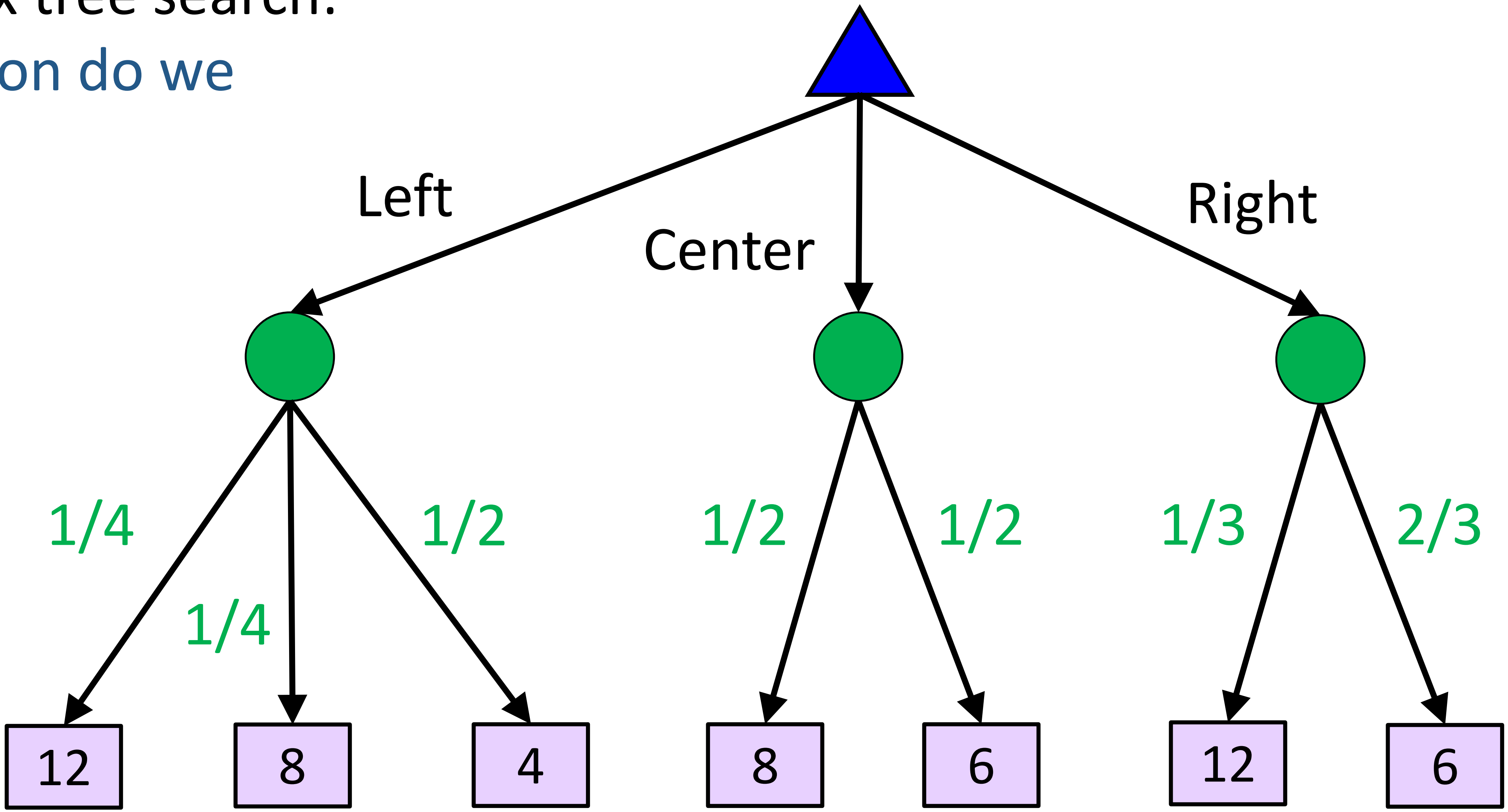
Which action do we choose?

A) Left

B) Center

C) Right

D) Eight



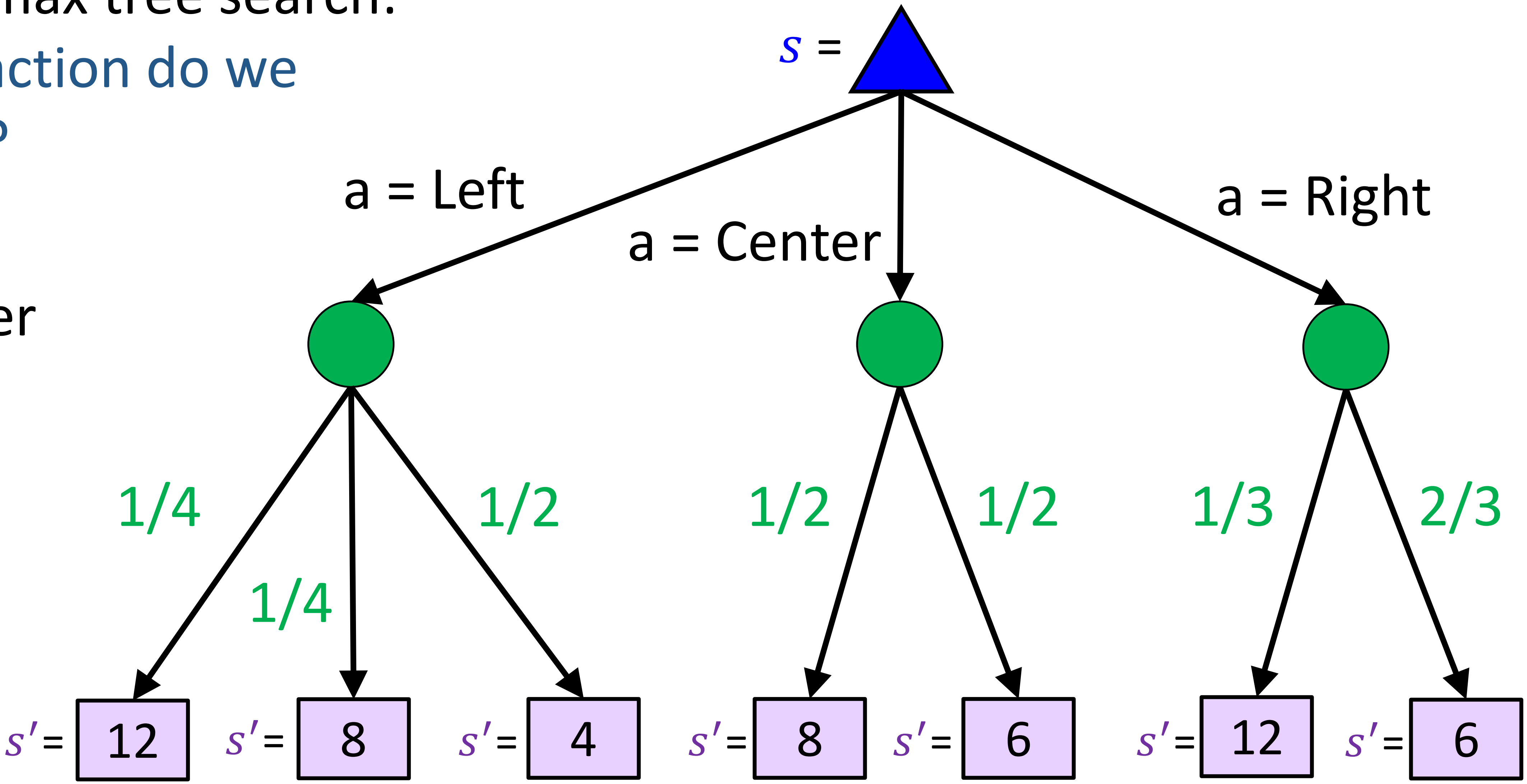
Piazza Poll 1

$$\hat{a} = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s'|s,a)\, V(s')$$

Expectimax tree search:

Which action do we choose?

A) Left
B) Center
C) Right
D) Eight



Piazza Poll 1

$$\hat{a} = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s'|s,a)\, V(s')$$

Expectimax tree search:

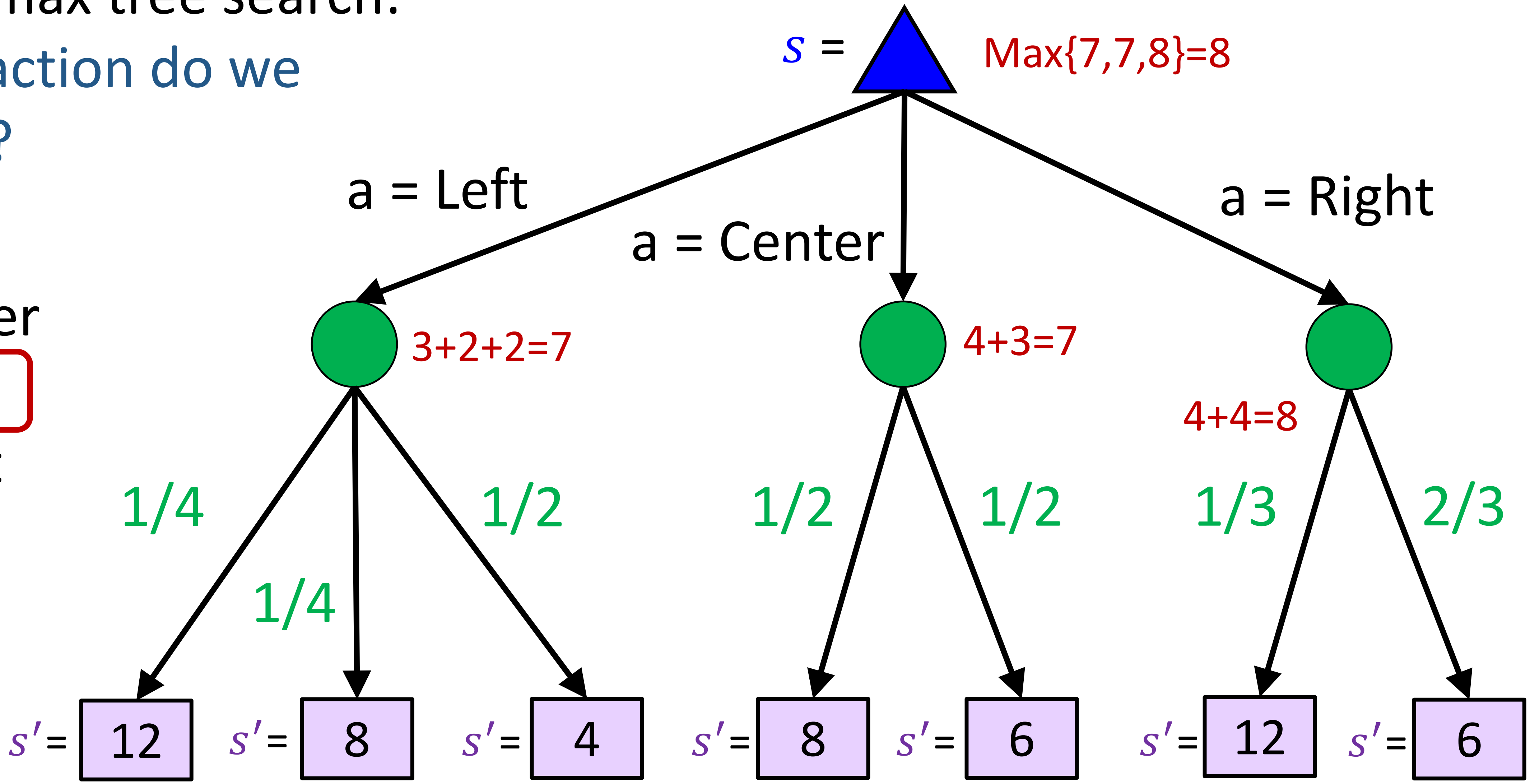
Which action do we choose?

A) Left

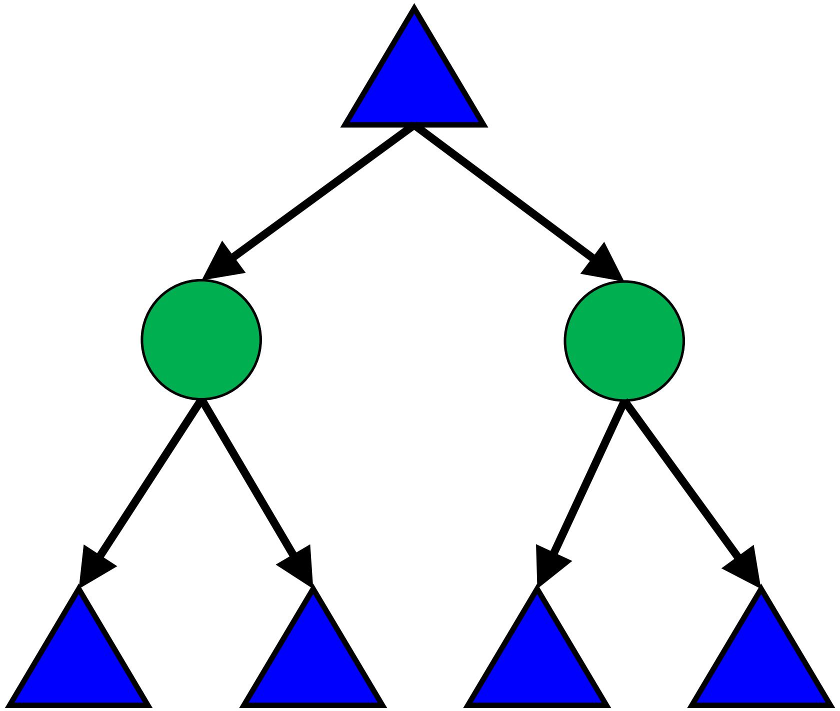
B) Center

C) Right

D) Eight



Expectimax Notation



$$V(s) = \max_a \sum_{s'} P(s'|s,a)\, V(s')$$

Warm-up as You Walk In

Given

- Set `actions` (persistent/static)
- Set `states` (persistent/static)
- Function `T(s,a,s_prime)`

Write the pseudo code for:

- `function V(s) return value`

that implements:

$$V(s) = \max_{a \in actions} \sum_{s' \in states} T(s, a, s')V(s')$$

MDP Notation

Standard expectimax: $V(s) = \max_a \sum_{s'} P(s'|s,a)V(s')$

Bellman equations: $V(s) = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')]$

Value iteration: $V_{k+1}(s) = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V_k(s')], \quad \forall\, s$

Q-iteration: $Q_{k+1}(s,a) = \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')], \quad \forall\, s,a$

Policy extraction: $\pi_V(s) = \operatorname{argmax}_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')], \quad \forall\, s$

Policy evaluation: $V_{k+1}^{\pi}(s) = \sum_{s'} P(s'|s,\pi(s))[R(s,\pi(s),s') + \gamma V_k^{\pi}(s')], \quad \forall\, s$

Policy improvement: $\pi_{new}(s) = \operatorname{argmax}_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V^{\pi_{old}}(s')], \quad \forall\, s$

MDP Notation

Standard expectimax: $V(s) = \max_a \sum_{s'} P(s'|s,a)V(s')$

Bellman equations: $V(s) = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')]$

Value iteration: $V_{k+1}(s) = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V_k(s')], \quad \forall\, s$

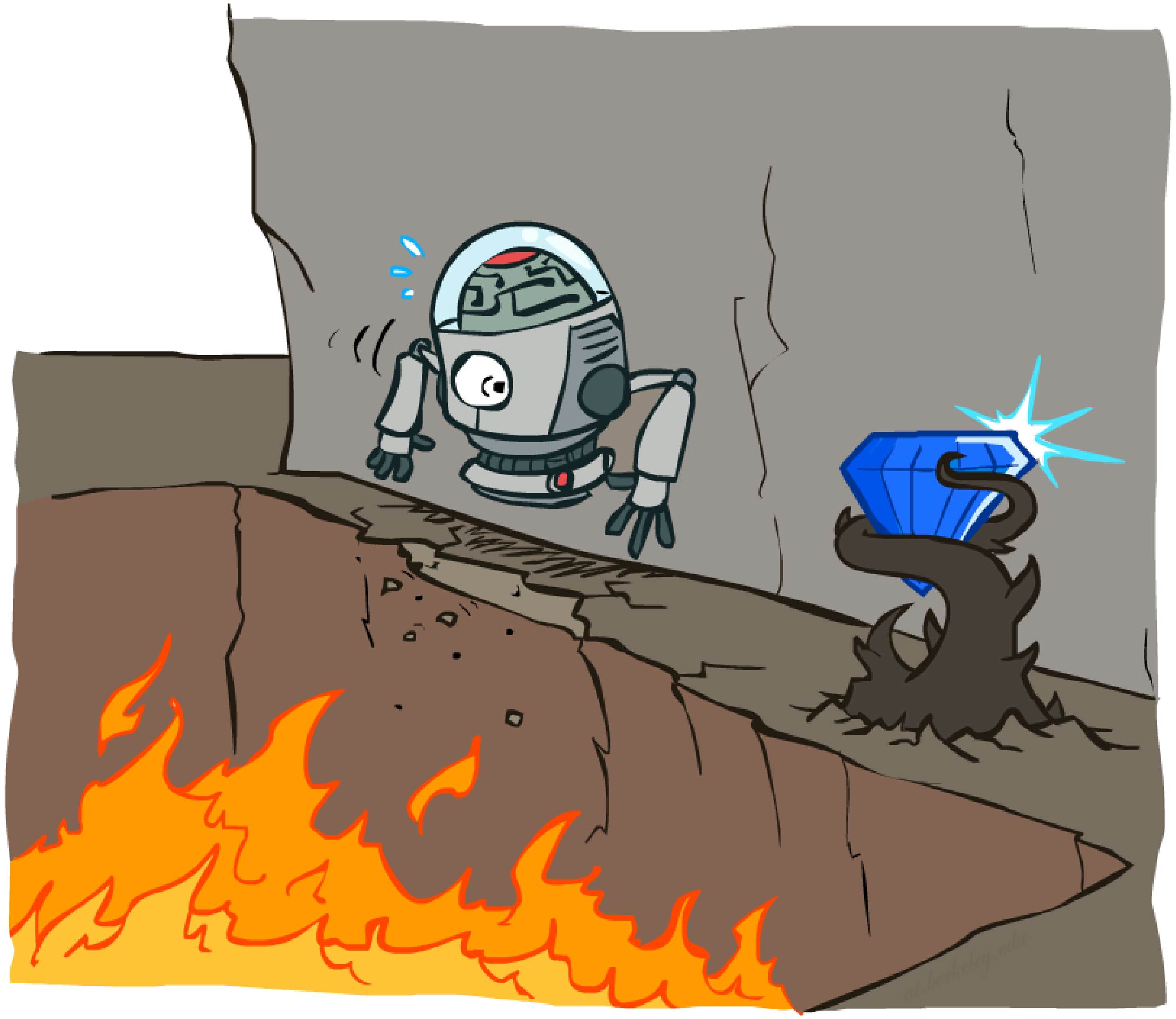
Q-iteration: $Q_{k+1}(s,a) = \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')], \quad \forall\, s, a$

Policy extraction: $\pi_V(s) = \operatorname{argmax}_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')], \quad \forall\, s$

Policy evaluation: $V_{k+1}^{\pi}(s) = \sum_{s'} P(s'|s,\pi(s))[R(s,\pi(s),s') + \gamma V_k^{\pi}(s')], \quad \forall\, s$

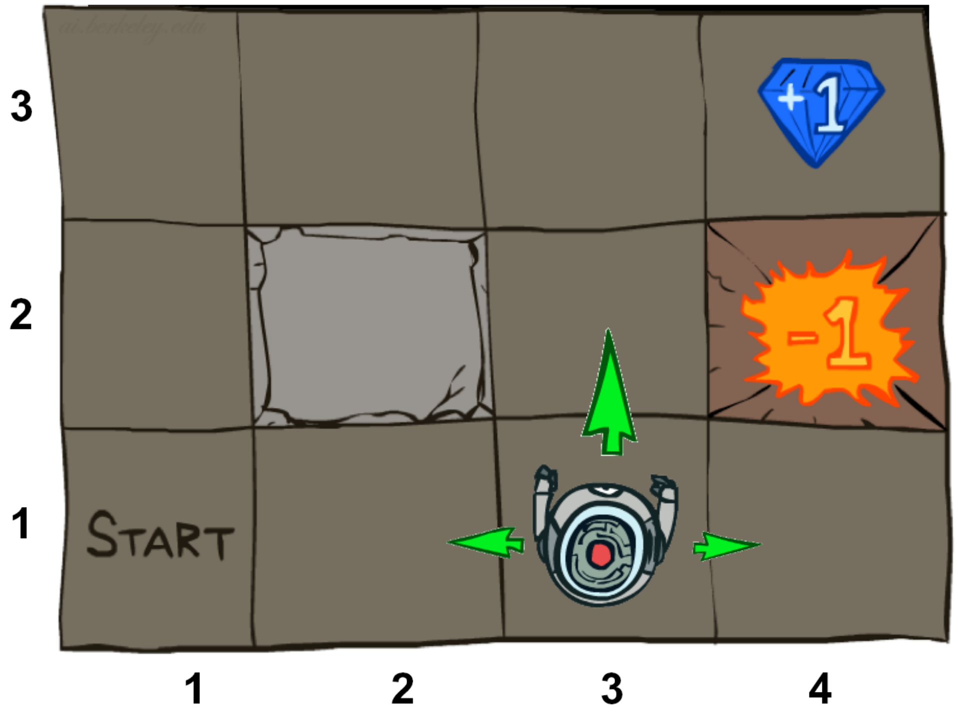
Policy improvement: $\pi_{new}(s) = \operatorname{argmax}_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V^{\pi_{old}}(s')], \quad \forall\, s$

Non-Deterministic Search



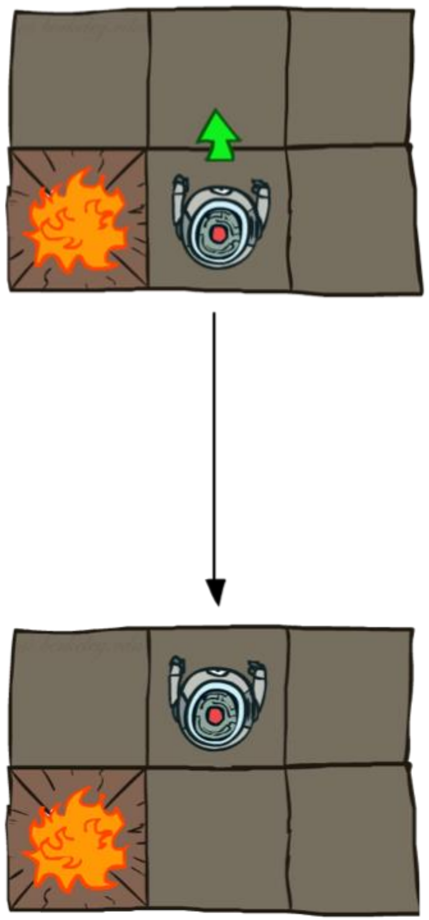
Example: Grid World

- A maze-like problem
  - The agent lives in a grid
  - Walls block the agent's path
- Noisy movement: actions do not always go as planned
  - If agent takes action North
    - 80% of the time: Get to the cell on the North (if there is no wall there)
    - 10%: West; 10%: East
  - If path after roll dice blocked by wall, stays put
- The agent receives rewards each time step
  - "Living" reward (can be negative)
  - Additional reward at pit or target (good or bad) and will exit the grid world afterward
- Goal: maximize sum of rewards

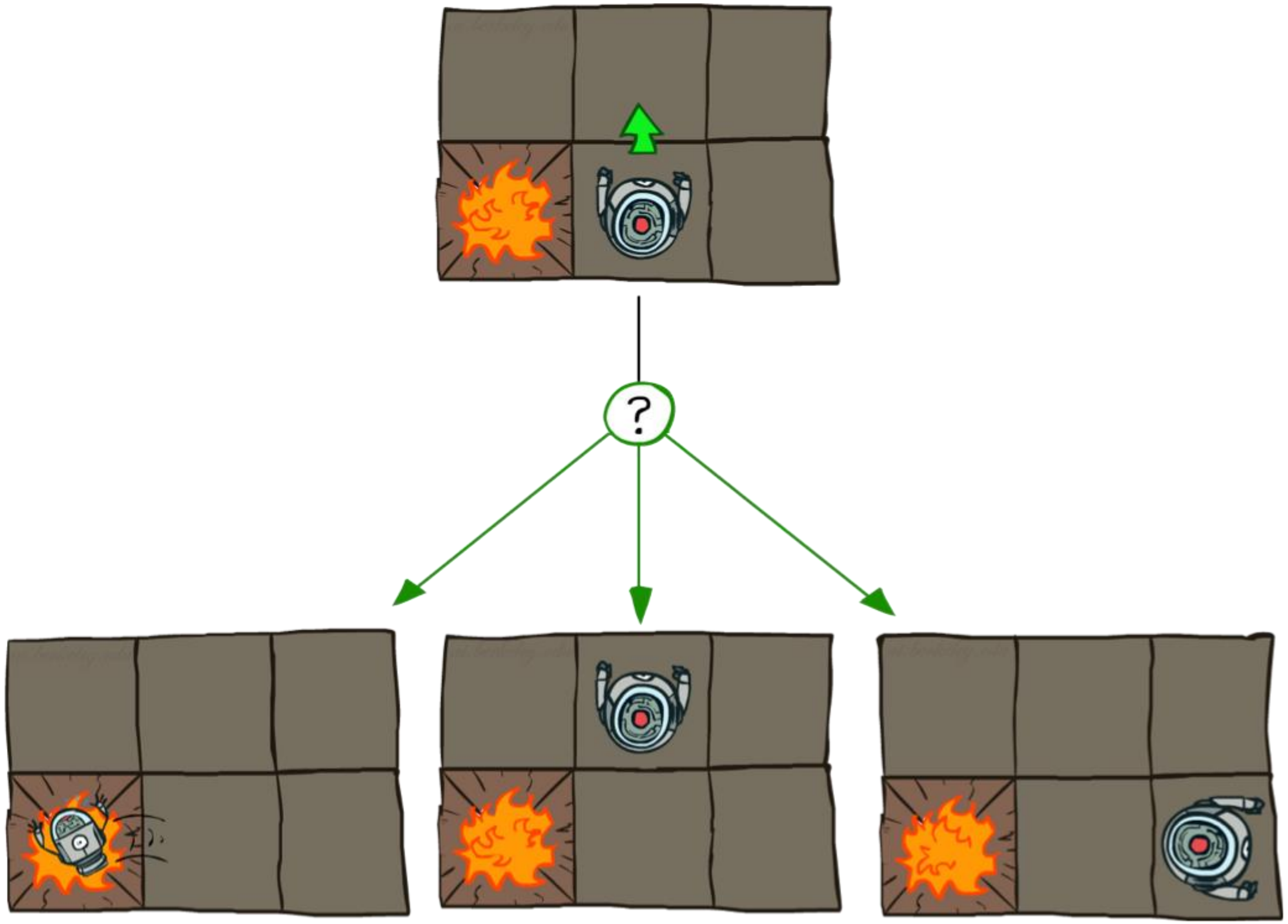


Grid World Actions

Deterministic Grid World



Stochastic Grid World

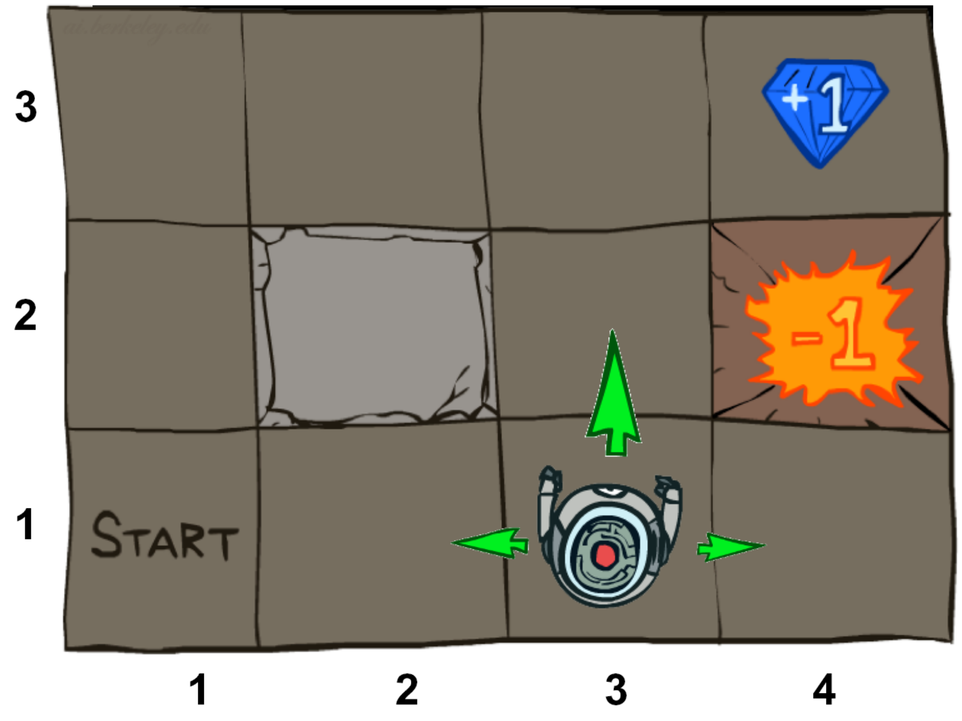


Markov Decision Processes

An MDP is defined by a tuple (S,A,T,R):

- S: a set of states
- A: a set of actions
- T: a transition function
  - T(s, a, s') where s ∈ S, a ∈ A, s' ∈ S is P(s' | s, a)
- R: a reward function
  - R(s, a, s') is reward at this time step
  - Sometimes just R(s) or R(s')

- Sometimes also have
  - γ: discount factor (introduced later)
  - μ: distribution of initial state (or just a start state s_0)
  - Terminal states: processes end after reaching these states



The Grid World problem as an MDP

R($s_{4,2}$, $exit$, $s_{virtual\_terminal}$)=-1

R($s_{4,2}$)=-1, no virtual terminal state

How to define the terminal states and reward function for the Grid World problem?

Demo of Gridworld

[Demo – gridworld manual intro (L8D1)]

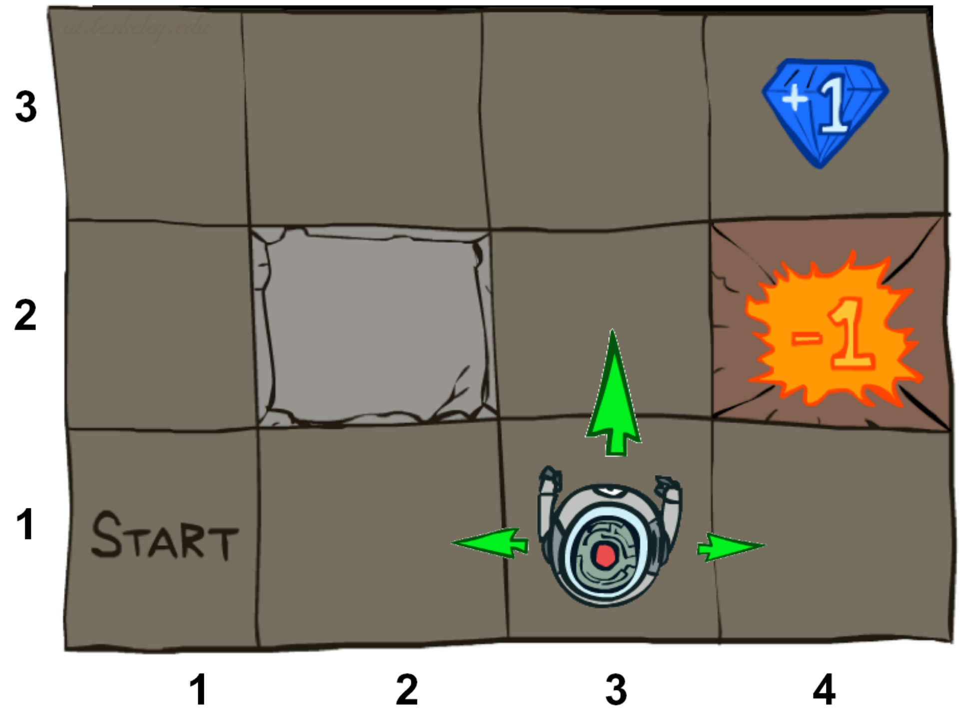
Markov Decision Processes

An MDP is defined by a tuple (S,A,T,R)

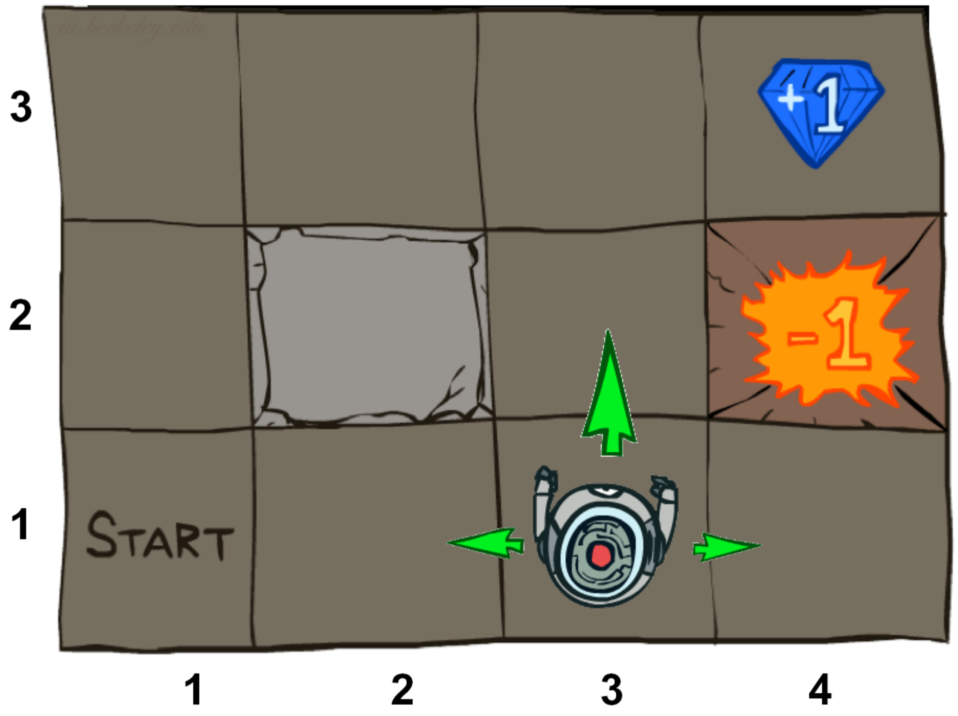
Why is it called Markov Decision Process?

Decision:

Process:



Markov Decision Processes



An MDP is defined by a tuple (S,A,T,R)

Why is it called Markov Decision Process?

Decision:

Agent decides what action to take in each time step

Process:

The system (environment + agent) is changing over time

What is Markov about MDPs?

Markov property: Conditional on the present state, the future and the past are independent

In MDP, it means outcome of an action depend only on current state

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t, S_{t-1} = s_{t-1}, A_{t-1}, \ldots S_0 = s_0)$$

$$=$$

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t)$$

Recall in search, successor function only depends on current state (not the history)

Andrey Markov
(1856-1922)
Russian mathematician

Policies

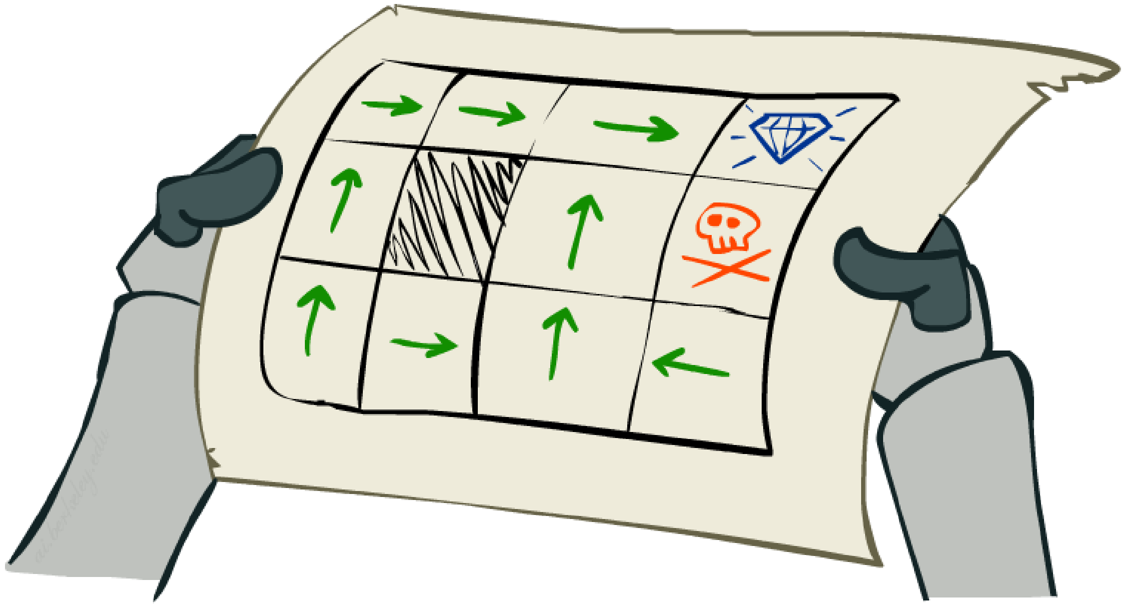
In deterministic single-agent search problems, we wanted an optimal plan, or sequence of actions, from start to a goal

For MDPs, we focus on policies

- Policy = map of states to actions
- $\pi(s)$ gives an action for state s

We want an optimal policy $\pi^*: S \rightarrow A$

- An optimal policy is one that maximizes expected utility if followed



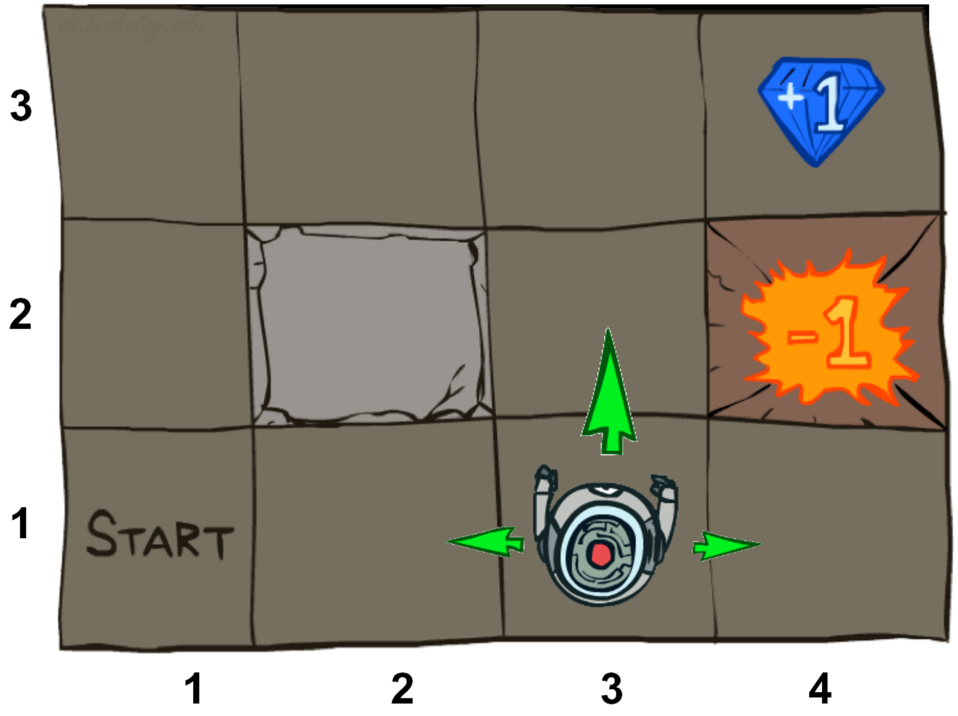
Policies

Recall: An MDP is defined S,A,T,R

Keep S,A,T fixed, optimal policy may vary given different R

What is the optimal policy if R(s,a,s')=-1000 for all states other than pit and target?

What is the optimal policy if R(s,a,s')=0 for all states other than pit and target, and reward=1000 and -1000 at pit and target respectively?



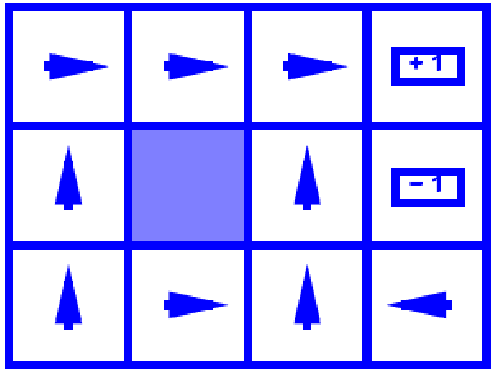
Piazza Poll 2

Which sequence of optimal policies matches the following sequence of living rewards:

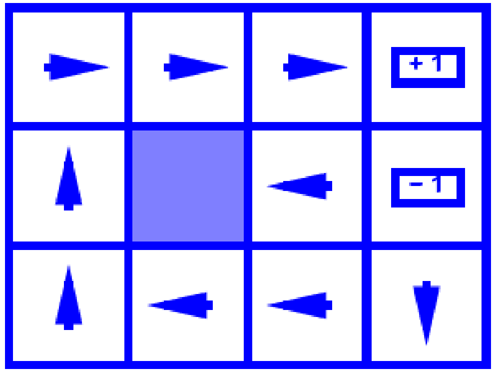
{-0.01, -0.03, -0.04, -2.0}

I. {B, A, C, D}
II. {B, C, A, D}
III. {C, B, A, D}
IV. {D, A, C, B}

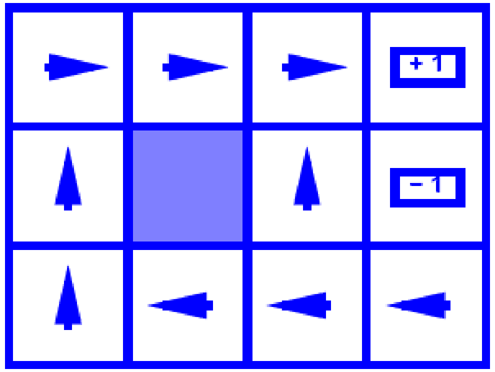
A)



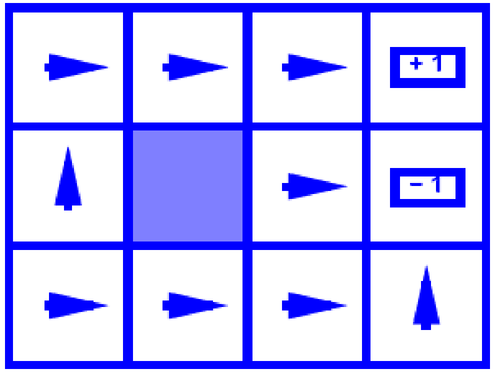
B)



C)



D)

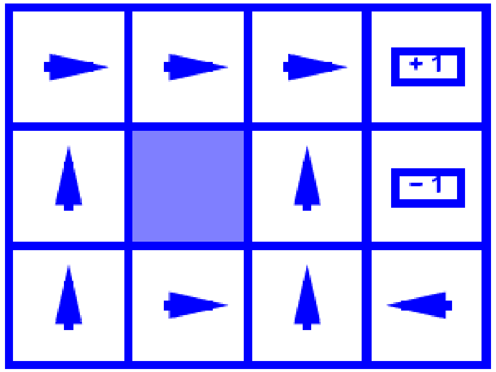


Piazza Poll 2

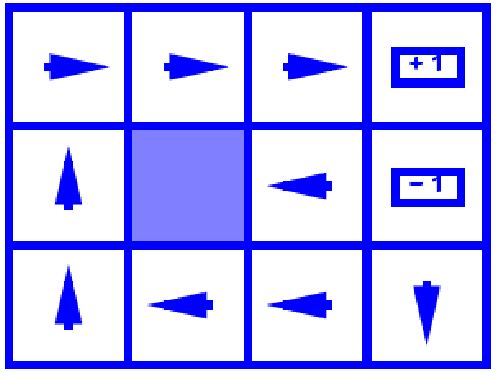
Which sequence of optimal policies matches the following sequence of living rewards: {-0.01, -0.03, -0.04, -2.0}

I. {B, A, C, D}
II. {B, C, A, D}
III. {C, B, A, D}
IV. {D, A, C, B}

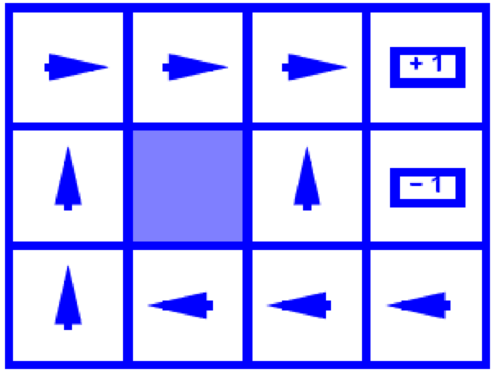
A)



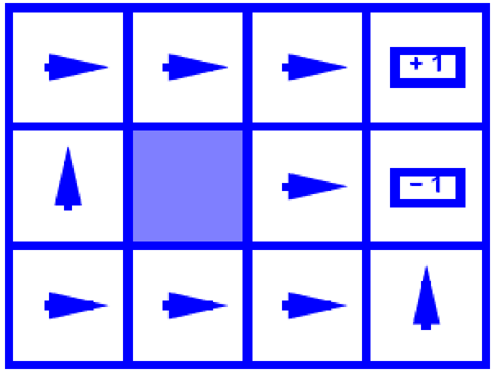
B)



C)

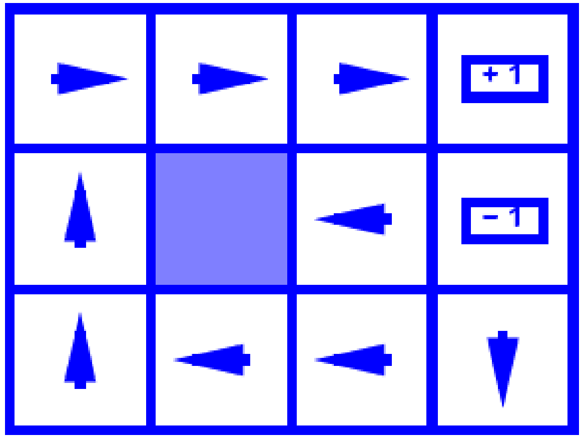


D)

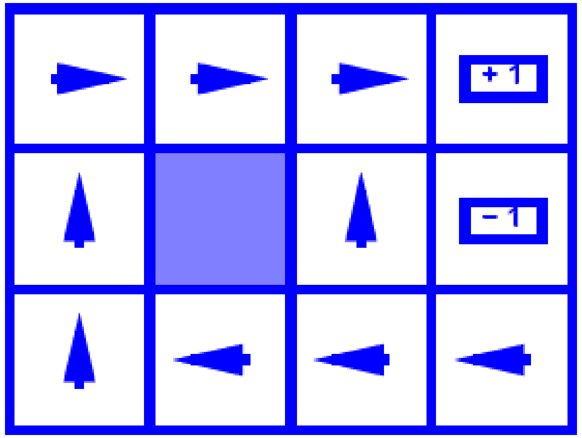


Piazza Poll 2

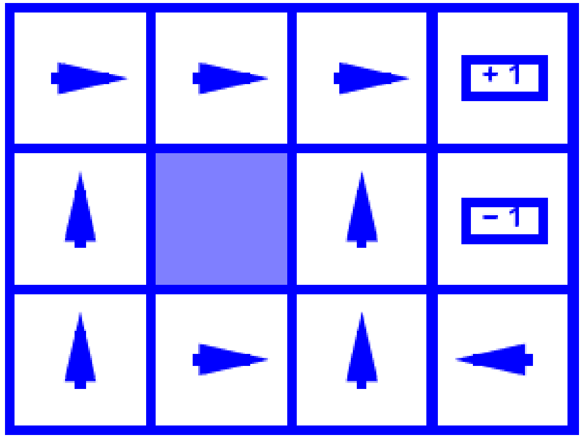
Optimal Policies



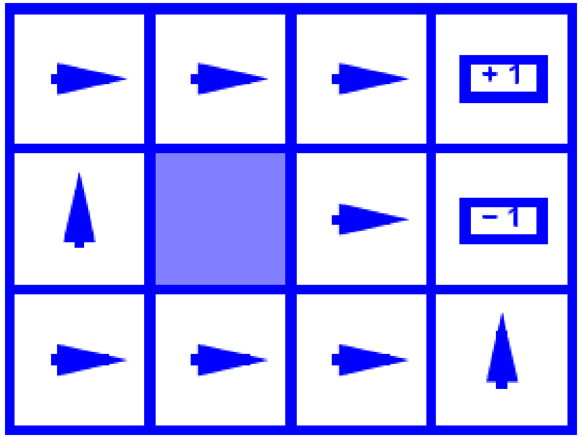
R(s) = -0.01



R(s) = -0.03

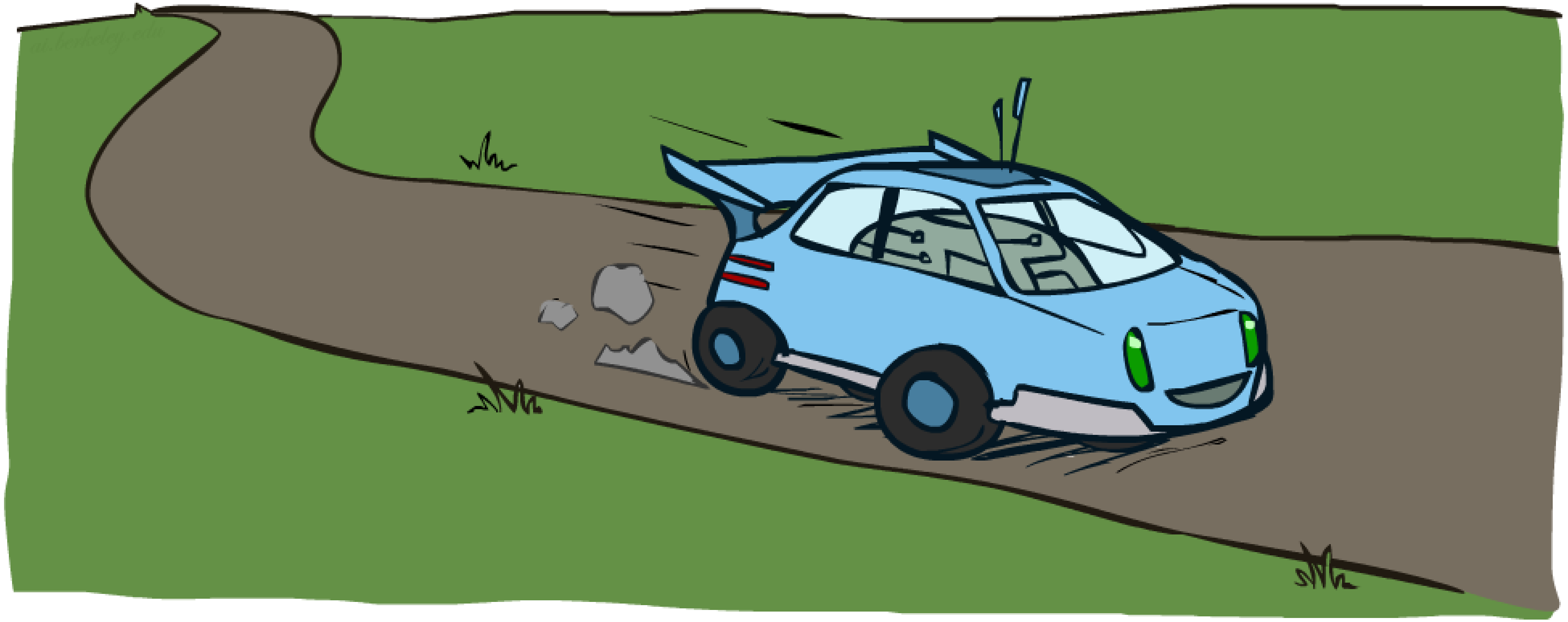


R(s) = -0.4



R(s) = -2.0

Example: Racing



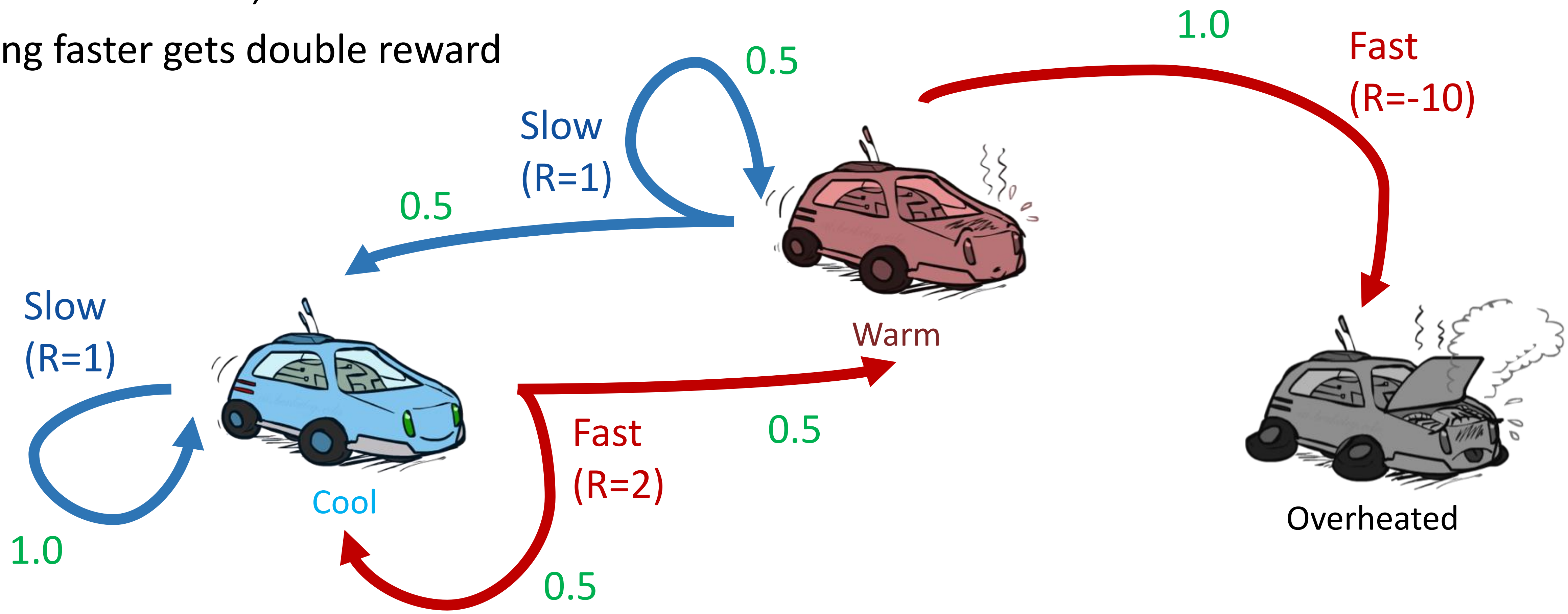
Example: Racing

A robot car wants to travel far, quickly

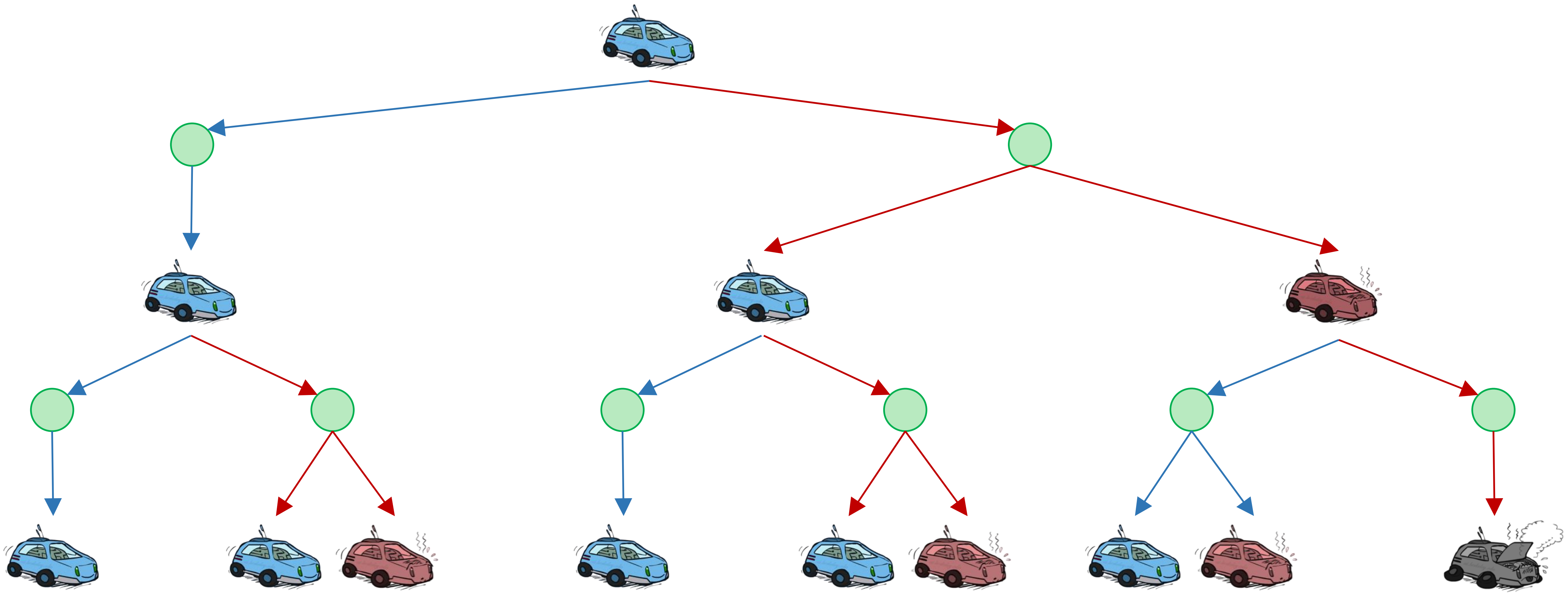
Three states: Cool, Warm, Overheated

Two actions: *Slow*, *Fast*

Going faster gets double reward



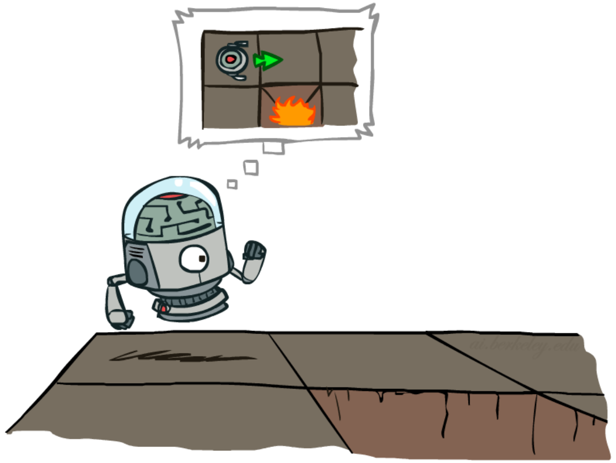
Racing Search Tree

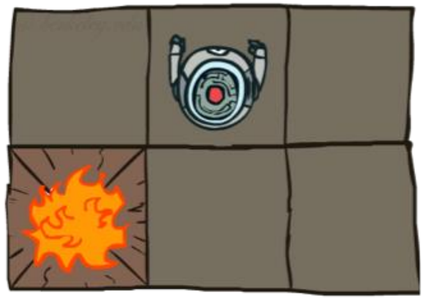


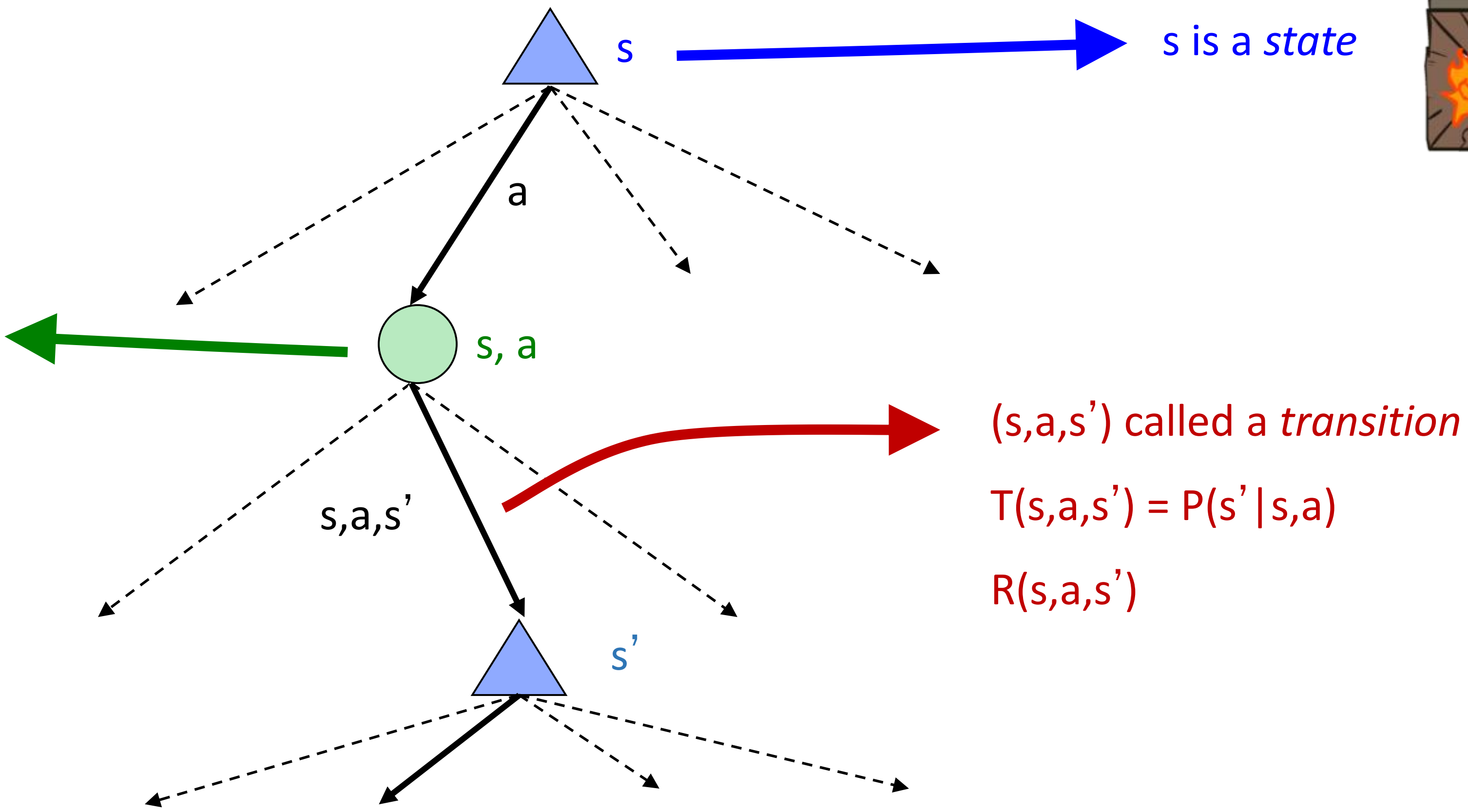
Can we use expectimax for MDP directly?

MDP Search Trees

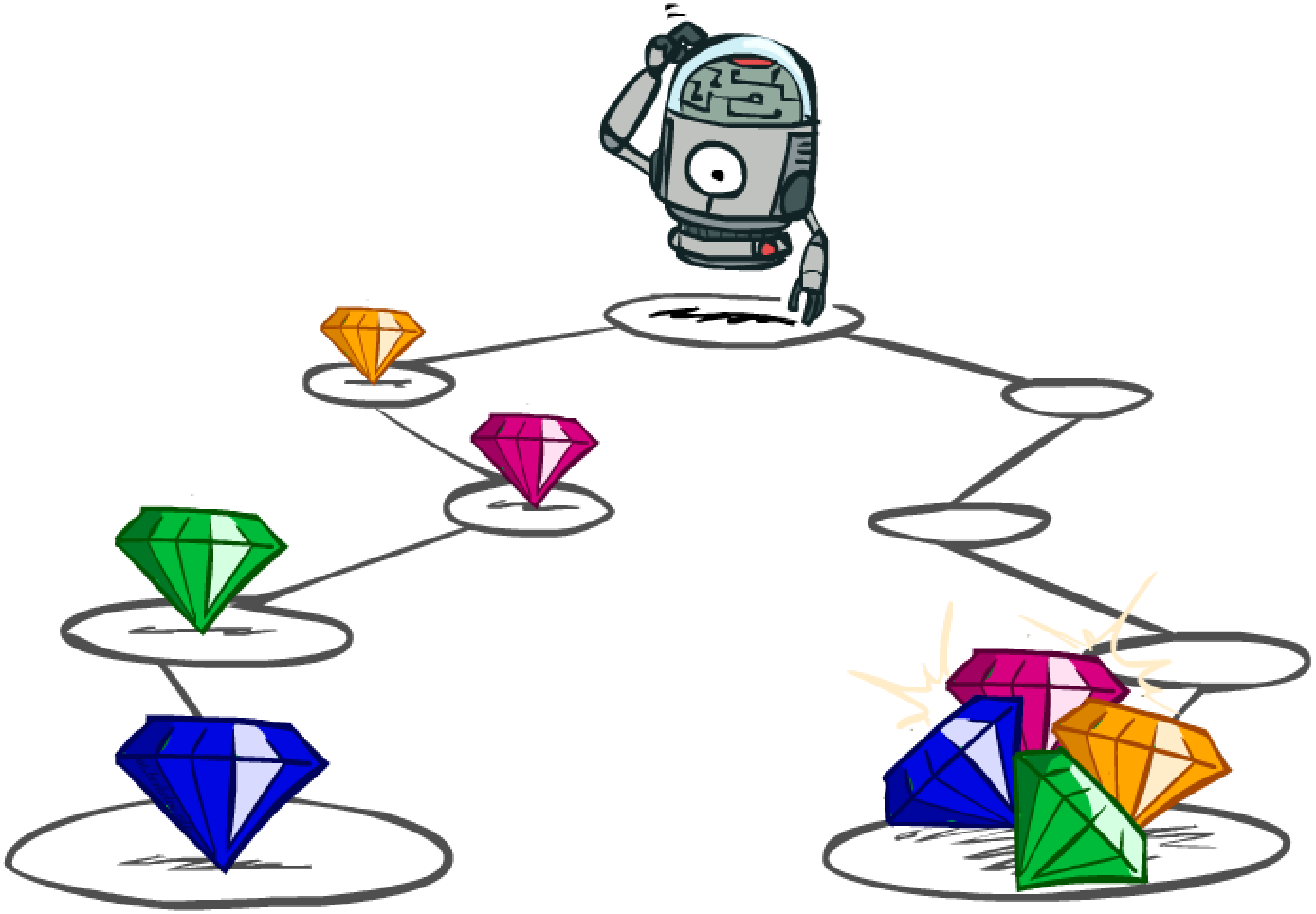
Each MDP state projects an expectimax-like search tree







Utilities of Sequences

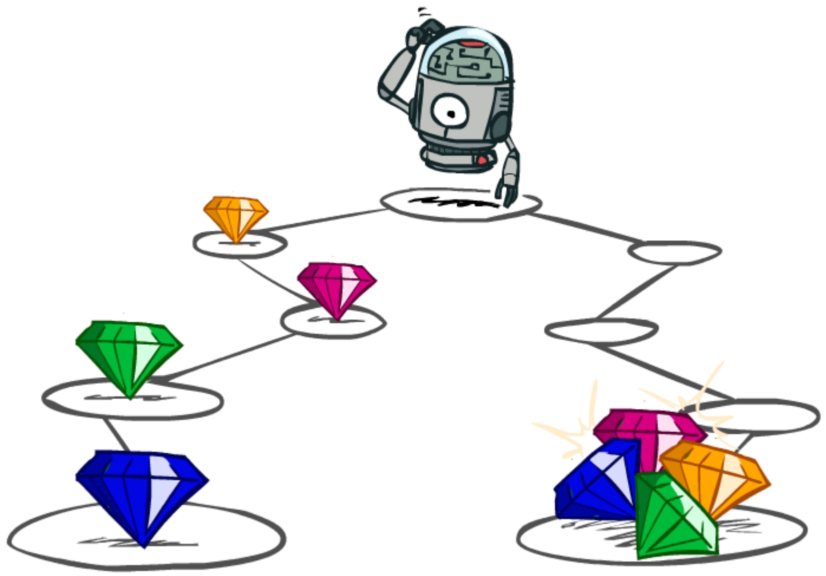


Utilities of Sequences

What preferences should an agent have over reward sequences?

More or less? [1, 2, 2] or [2, 3, 4]

Now or later? [0, 0, 1] or [1, 0, 0]



Discounting

It's reasonable to maximize the sum of rewards

It's also reasonable to prefer rewards now to rewards later

One solution: utility of rewards decay exponentially

1

Worth Now

γ

Worth Next Step

γ^2

Worth In Two Steps

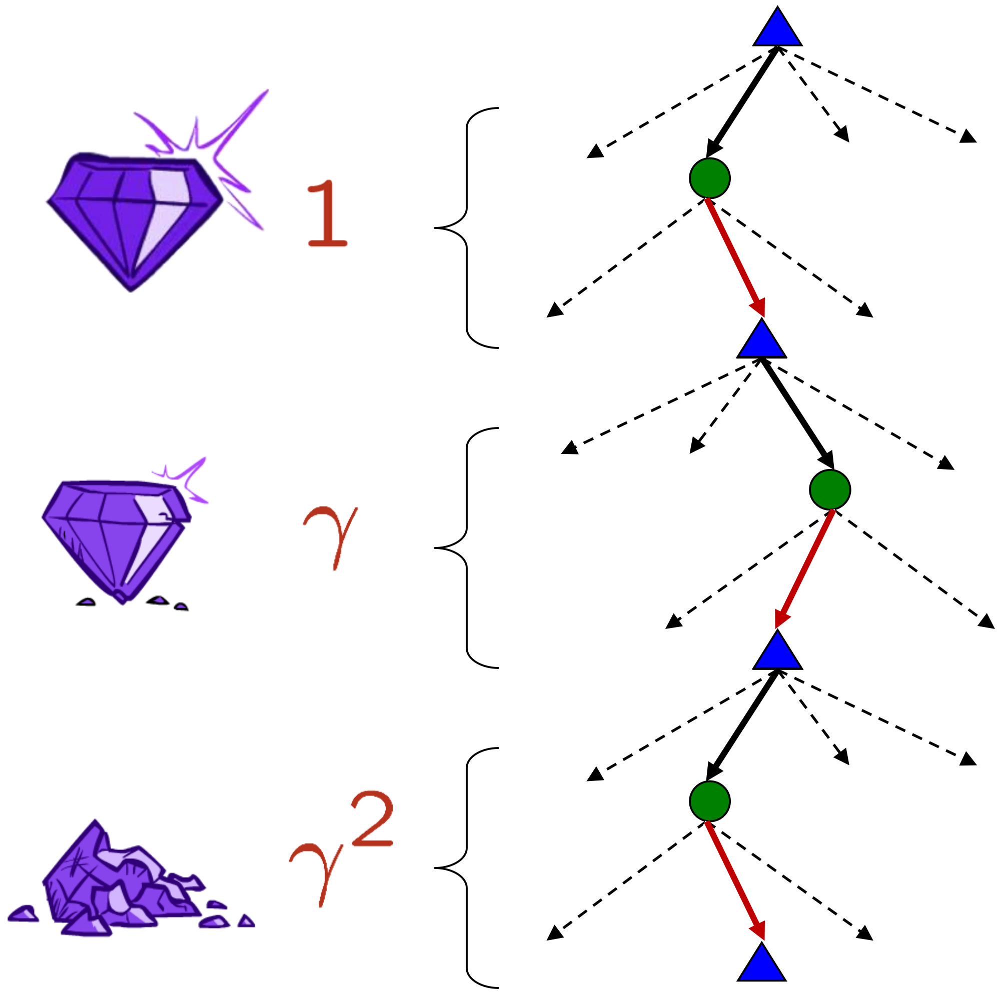
Discounting

How to discount?

- Each time we descend a level, we multiply in the discount once

Why discount?

- Sooner rewards probably do have higher utility than later rewards
- Also helps our algorithms converge



Piazza Poll 3

What is the value of U([2,4,8]) with $\gamma = 0.5$?

U(·) is the total utility of a reward sequence

A. 3
B. 6
C. 7
D. 14

Bonus: What is the value of U([8,4,2]) with $\gamma = 0.5$?

Piazza Poll 3

What is the value of U([2,4,8]) with $\gamma = 0.5$?

U(·) is the total utility of a reward sequence

A. 3
B. 6
C. 7
D. 14

$$\gamma^0 \times 2 + \gamma^1 \times 4 + \gamma^2 \times 8 = 2 + 0.5 \times 4 + 0.5 \times 0.5 \times 8 = 2 + 2 + 2 = 6$$

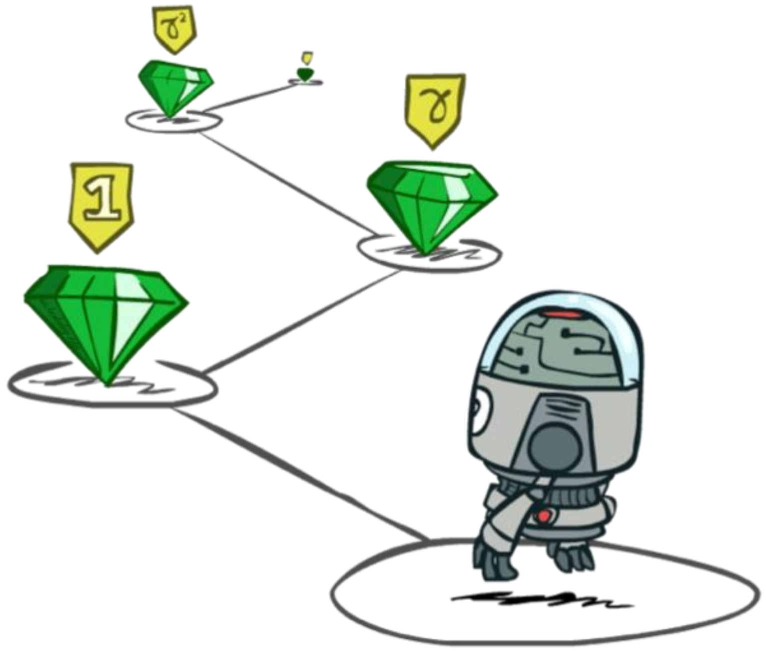
Bonus: What is the value of U([8,4,2]) with $\gamma = 0.5$?

$$\gamma^0 \times 8 + \gamma^1 \times 4 + \gamma^2 \times 2 = 8 + 0.5 \times 4 + 0.5 \times 0.5 \times 2 = 8 + 2 + 0.5 = 10.5$$

Stationary Preferences

Theorem: if we assume stationary preferences:

$$[a_1, a_2, \ldots] \succ [b_1, b_2, \ldots]$$
$$\Updownarrow$$
$$[r, a_1, a_2, \ldots] \succ [r, b_1, b_2, \ldots]$$

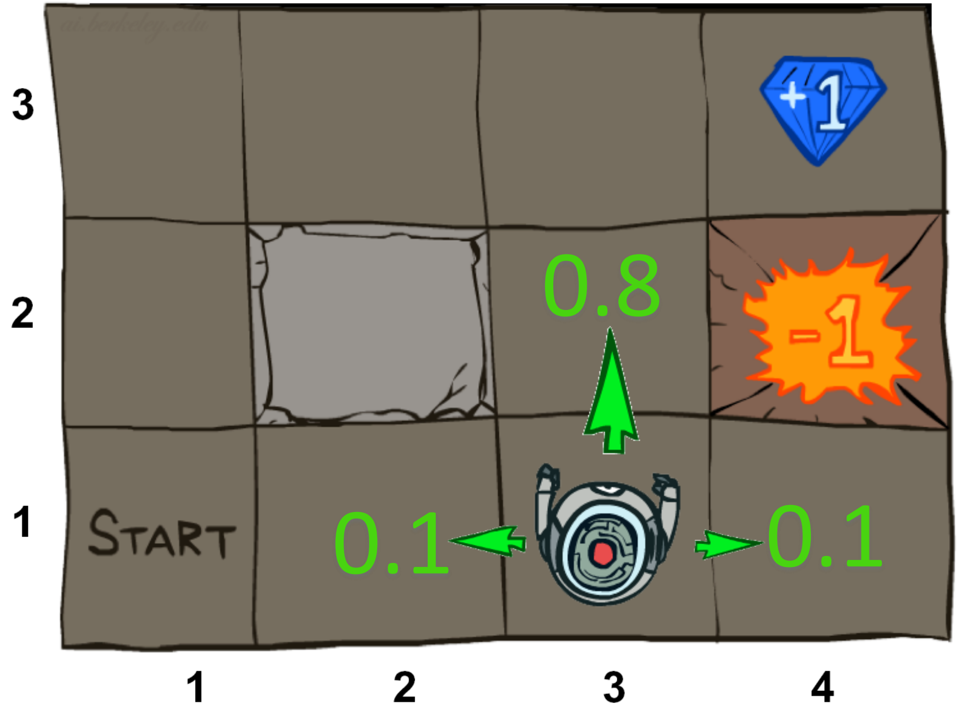


Then: there are only two ways to define utilities

- Additive utility: $U([r_0, r_1, r_2, \ldots]) = r_0 + r_1 + r_2 + \cdots$
- Discounted utility: $U([r_0, r_1, r_2, \ldots]) = r_0 + \gamma r_1 + \gamma^2 r_2 \cdots$

Question

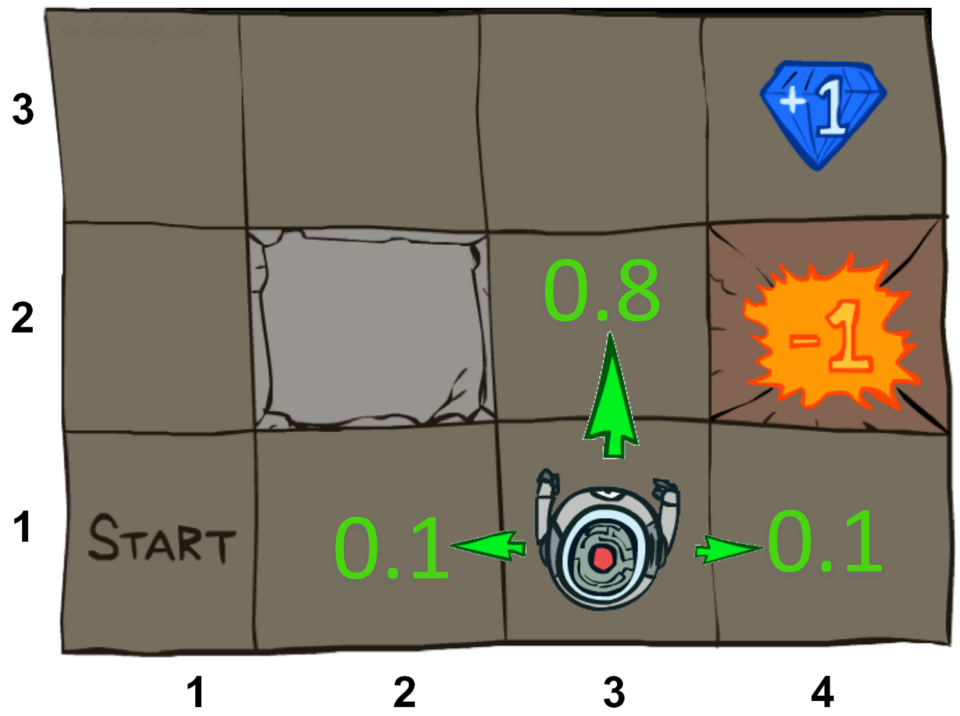
What is the minimum and maximum possible length of reward sequence in this grid world problem?



Question

What is the minimum and maximum possible length of reward sequence in this grid world problem?

5, $+\infty$

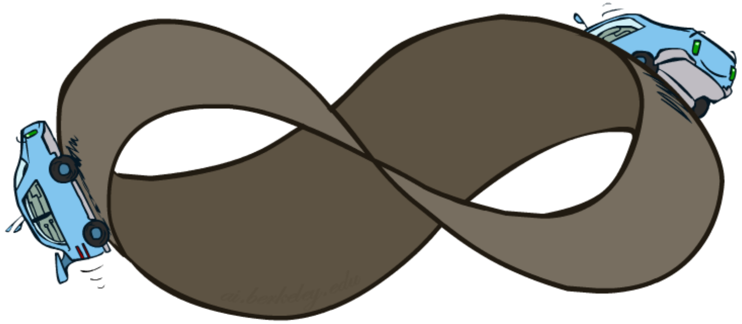


Infinite Utilities?!

What if the sequence is infinite? Do we get infinite utility?

- With discounting γ where $0 < \gamma < 1$ Assume $|r_t| \leq R_{max}$

$$U([r_0, \ldots r_\infty]) = \sum_{t=0}^{\infty} \gamma^t r_t \leq R_{\max}/(1-\gamma)$$



- Smaller γ means smaller "horizon" – shorter term focus

Optimal Policy with Discounting

Given:

10				1
a	b	c	d	e

- Actions: East, West, and Exit (only available in exit states a, e)
- Transitions: deterministic

For $\gamma = 1$, what is the optimal policy?

10				1

For $\gamma = 0.1$, what is the optimal policy?

10				1

For which γ are West and East equally good when in state d?

Optimal Policy with Discounting

Given:

a	b	c	d	e
10				1

- Actions: East, West, and Exit (only available in exit states a, e)
- Transitions: deterministic

For $\gamma = 1$, what is the optimal policy?

10	←	←	←	1

For $\gamma = 0.1$, what is the optimal policy?

10	←	←	→	1

For which γ are West and East equally good when in state d?

$$\gamma^3 \times 10 = \gamma^1 \times 1$$

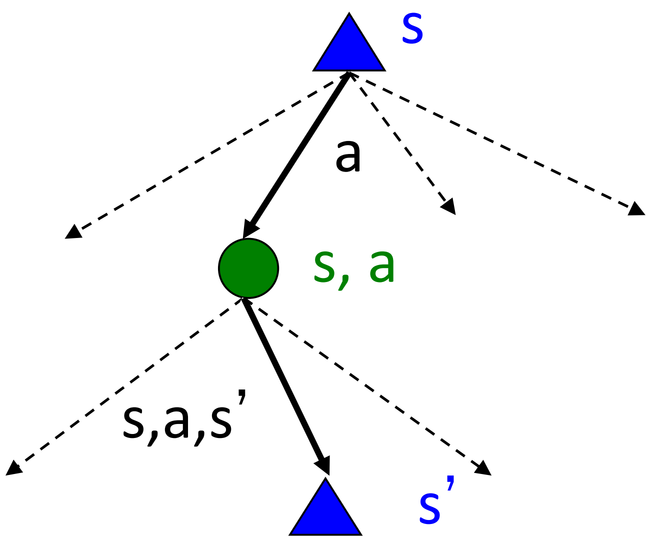
MDP Quantities

Markov decision processes:

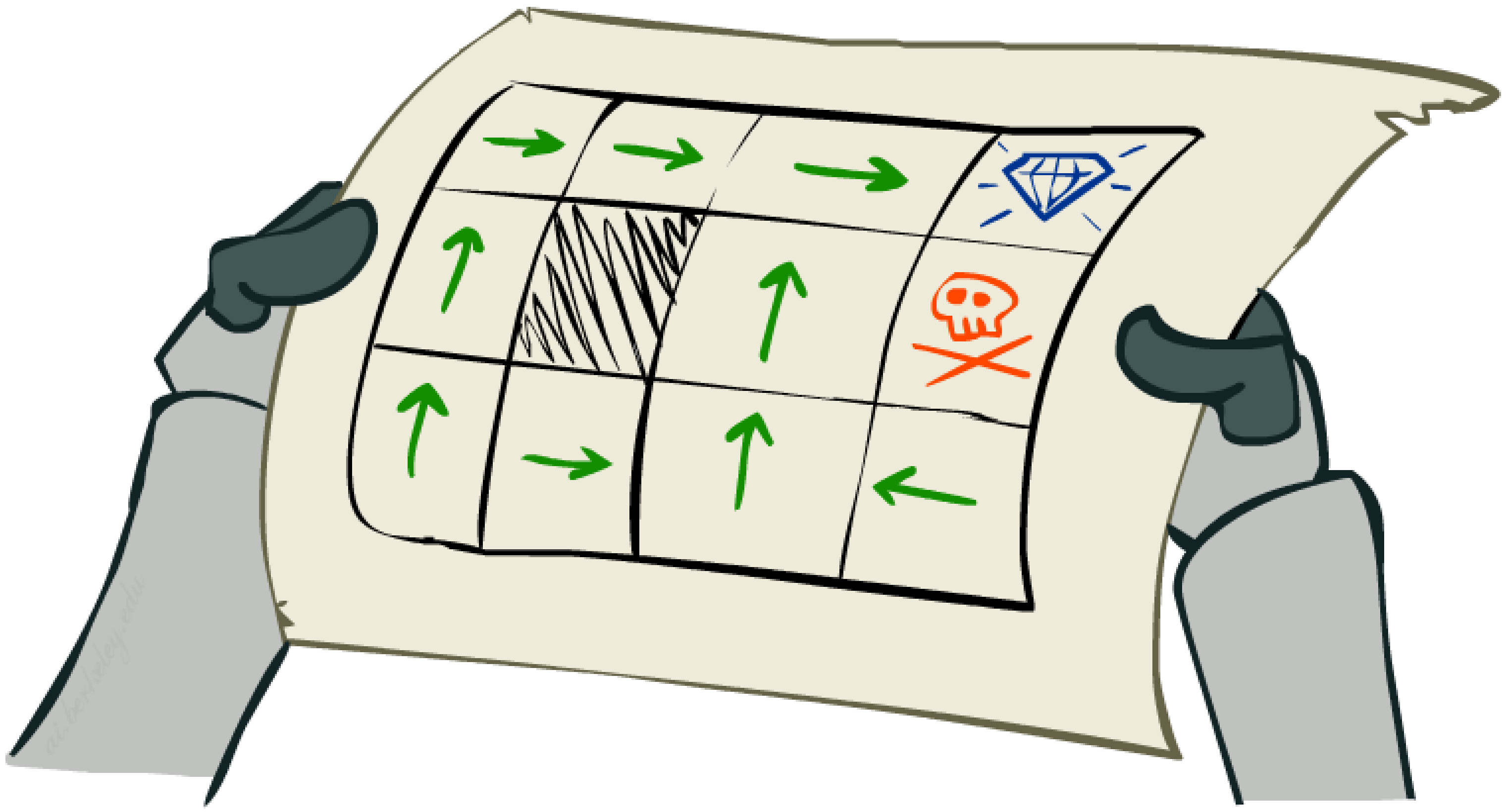
- States S
- Actions A
- Transitions P(s'|s,a) (or T(s,a,s'))
- Rewards R(s,a,s') (and discount γ)
- Start state s_0

MDP quantities so far:

- Policy = map of states to actions
- Utility = sum of (discounted) rewards



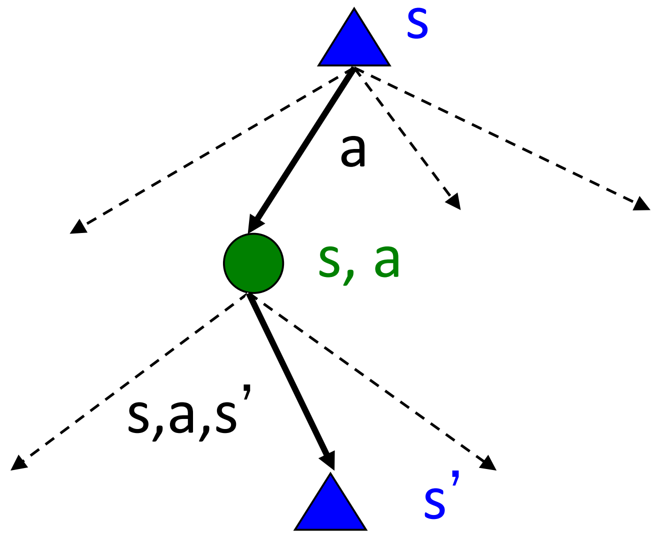
Solving MDPs



MDP Quantities

Markov decision processes:

- States S
- Actions A
- Transitions P(s'|s,a) (or T(s,a,s'))
- Rewards R(s,a,s') (and discount γ)
- Start state s_0

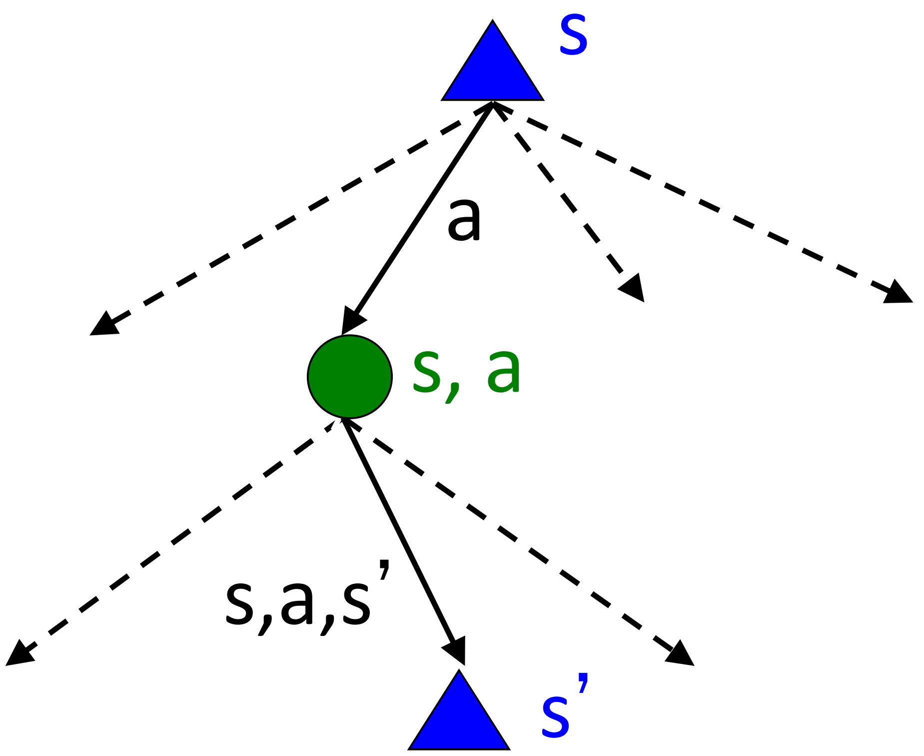


MDP quantities:

- Policy = map of states to actions
- Utility = sum of (discounted) rewards
- (State) Value = expected utility starting from a state (max node)
- Q-Value = expected utility starting from a state-action pair, i.e., q-state (chance node)

MDP Optimal Quantities

- The optimal policy:
  $\pi^*(s)$ = optimal action from state s
- The (true) value (or utility) of a state s:
  $V^*(s)$ = expected utility starting in s and acting optimally
- The (true) value (or utility) of a q-state (s,a):
  $Q^*(s,a)$ = expected utility starting out having taken action a from state s and (thereafter) acting optimally



Solve MDP: Find π^*, V^* and/or Q^*

[Demo: gridworld values (L9D1)]

Snapshot of Demo – Gridworld V Values

What is the optimal policy?

Noise = 0.2
Discount = 0.9
Living reward = 0

Snapshot of Demo – Gridworld V Values

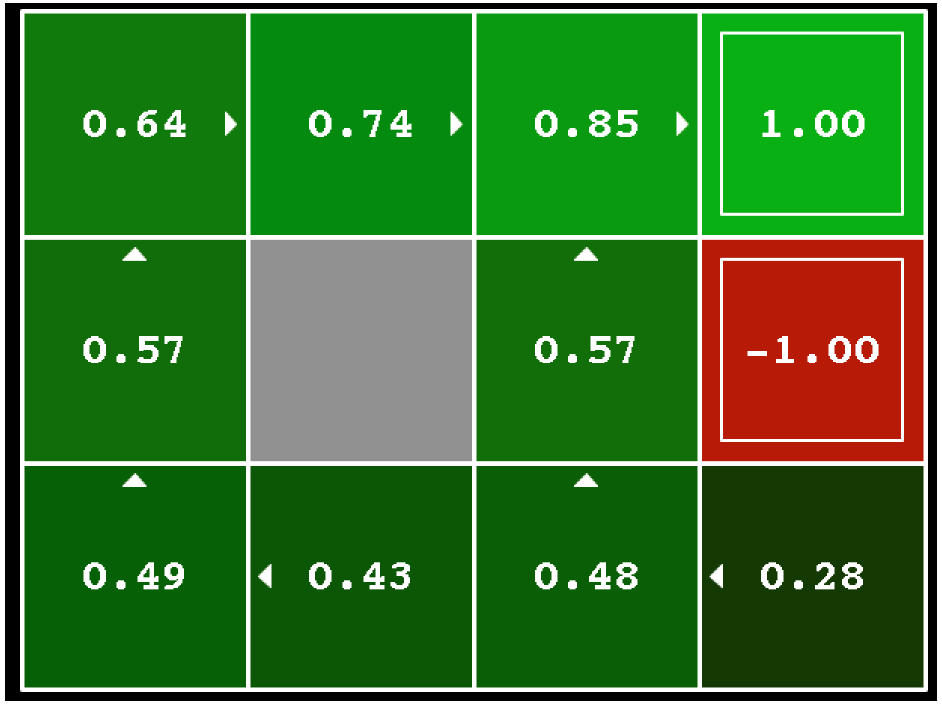
What is the
optimal policy?

May not equal $\underset{a}{\operatorname{argmax}} V^*(s')$ where s' is the most likely state after taking action a

$V^*(s)$ = expected utility starting in s and acting optimally

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s'|s,a) * \underbrace{(R(s,a,s')}_{=0} + \gamma V^*(s'))$$

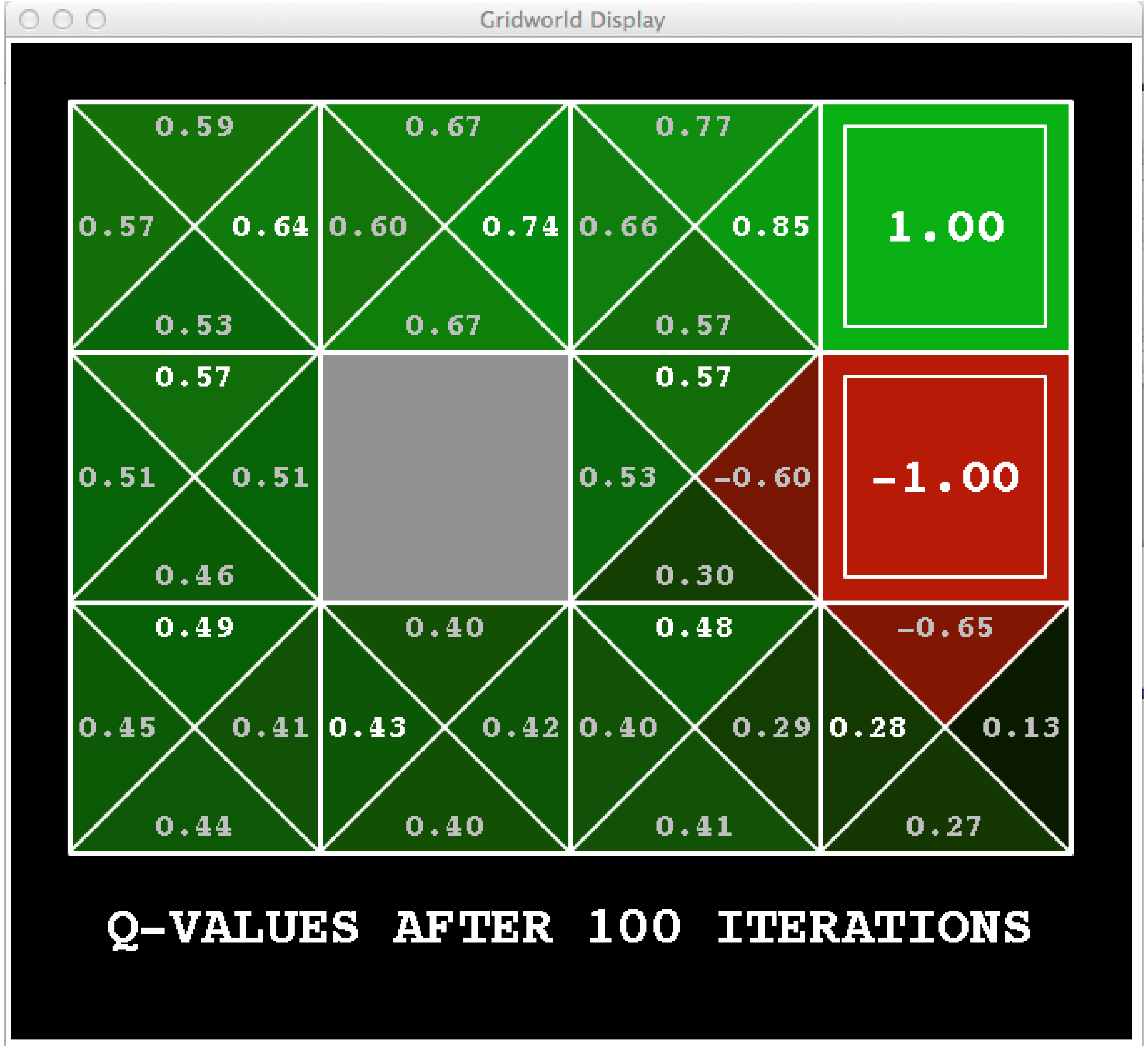
$$= \underset{a}{\operatorname{argmax}} \sum_{s'} P(s'|s,a) V^*(s'))$$



Noise = 0.2
Discount = 0.9
Living reward = 0

Snapshot of Demo – Gridworld Q Values

What is the optimal policy?



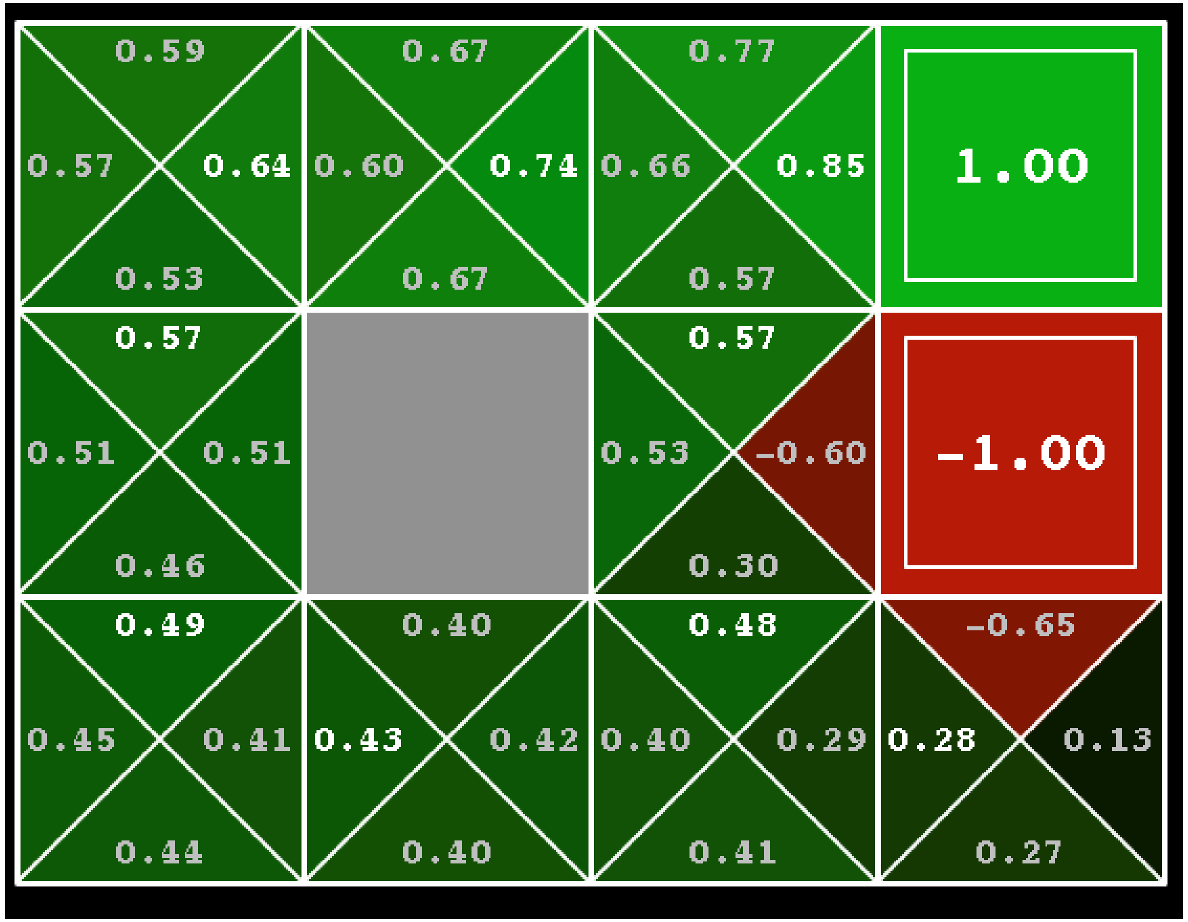
Noise = 0.2
Discount = 0.9
Living reward = 0

Snapshot of Demo – Gridworld Q Values

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q^*(s,a)$$

What is the optimal policy?

$Q^*(s,a)$ = expected utility starting out having taken action a from state s and (thereafter) acting optimally



Noise = 0.2
Discount = 0.9
Living reward = 0

Snapshot of Demo – Gridworld V Values

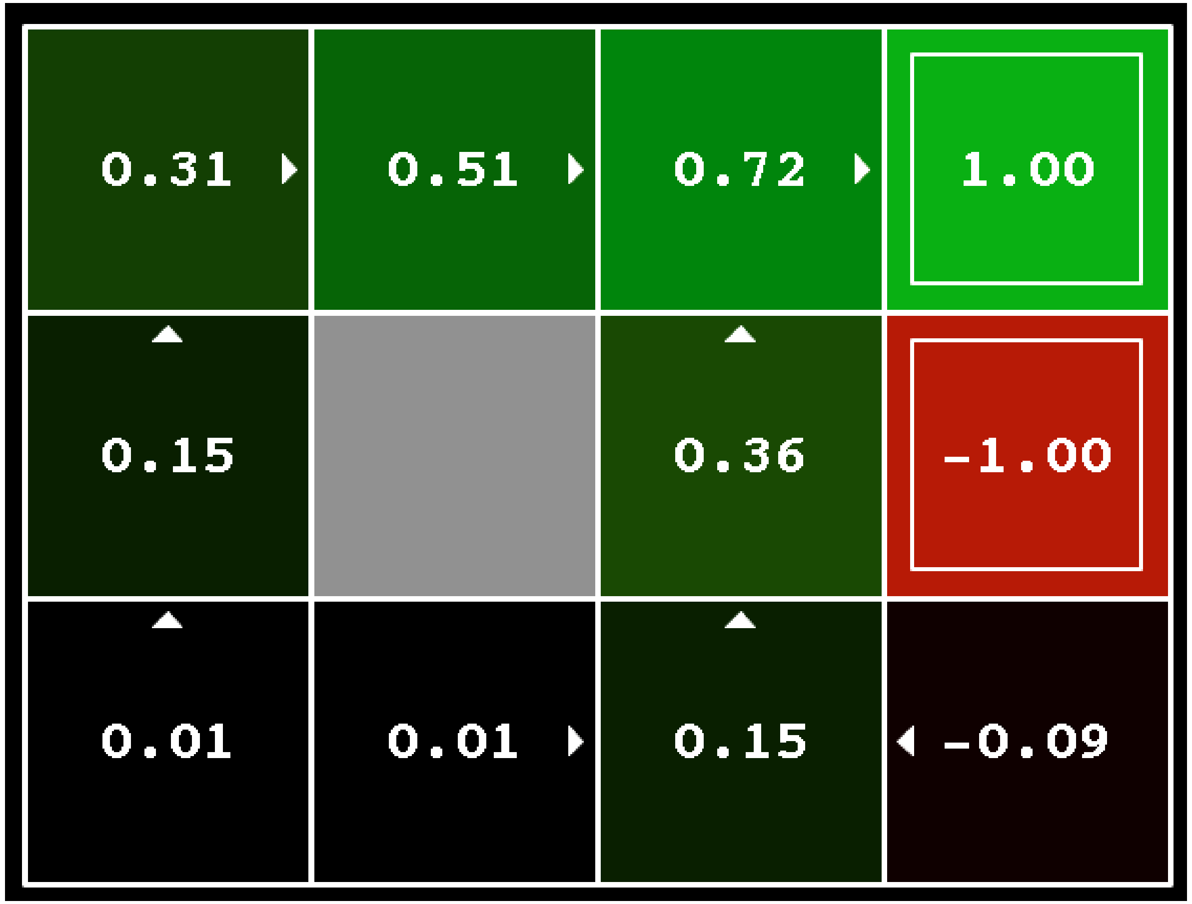
What is the optimal policy?

Noise = 0.2
Discount = 0.9
Living reward = -0.1

Snapshot of Demo – Gridworld V Values

$$\pi^*(s) = \underset{a}{\text{argmax}} \sum_{s'} P(s'|s,a) * (R(s,a,s') + \gamma V^*(s')) \neq \underset{a}{\text{argmax}}\, V^*(s')$$

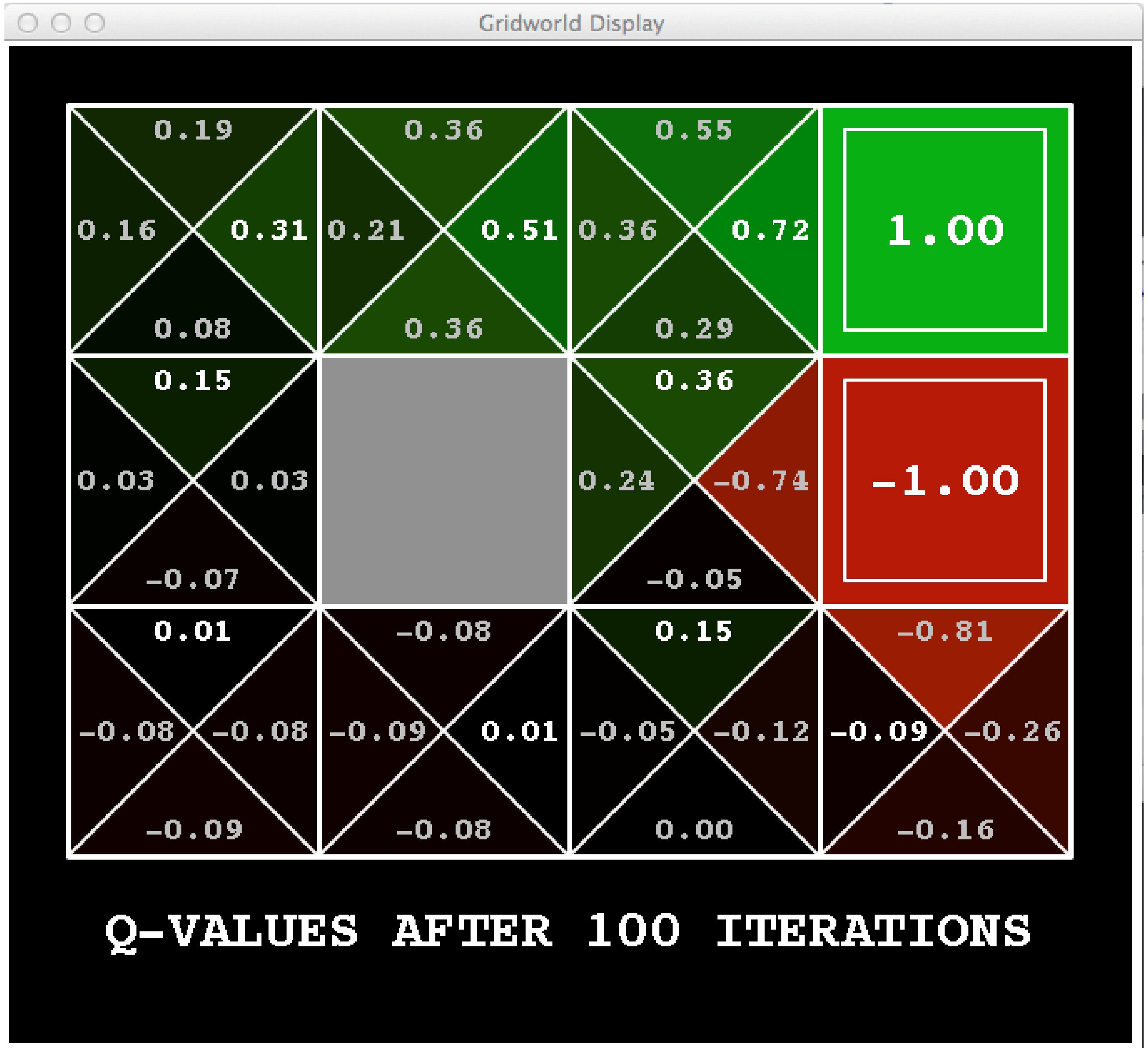
What is the optimal policy?



Noise = 0.2
Discount = 0.9
Living reward = -0.1

Snapshot of Demo – Gridworld Q Values

What is the optimal policy?

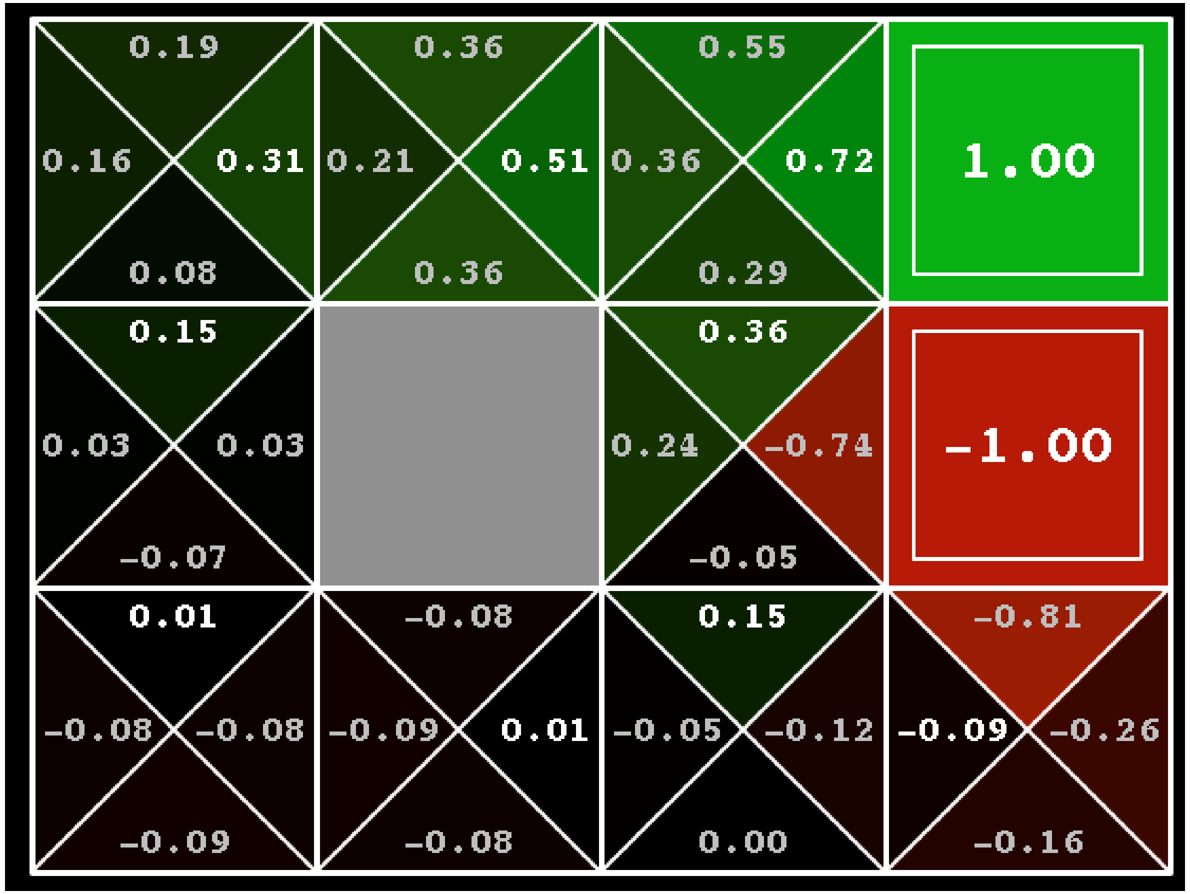


Noise = 0.2
Discount = 0.9
Living reward = -0.1

Snapshot of Demo – Gridworld Q Values

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q^*(s,a)$$

What is the optimal policy?



Noise = 0.2
Discount = 0.9
Living reward = -0.1

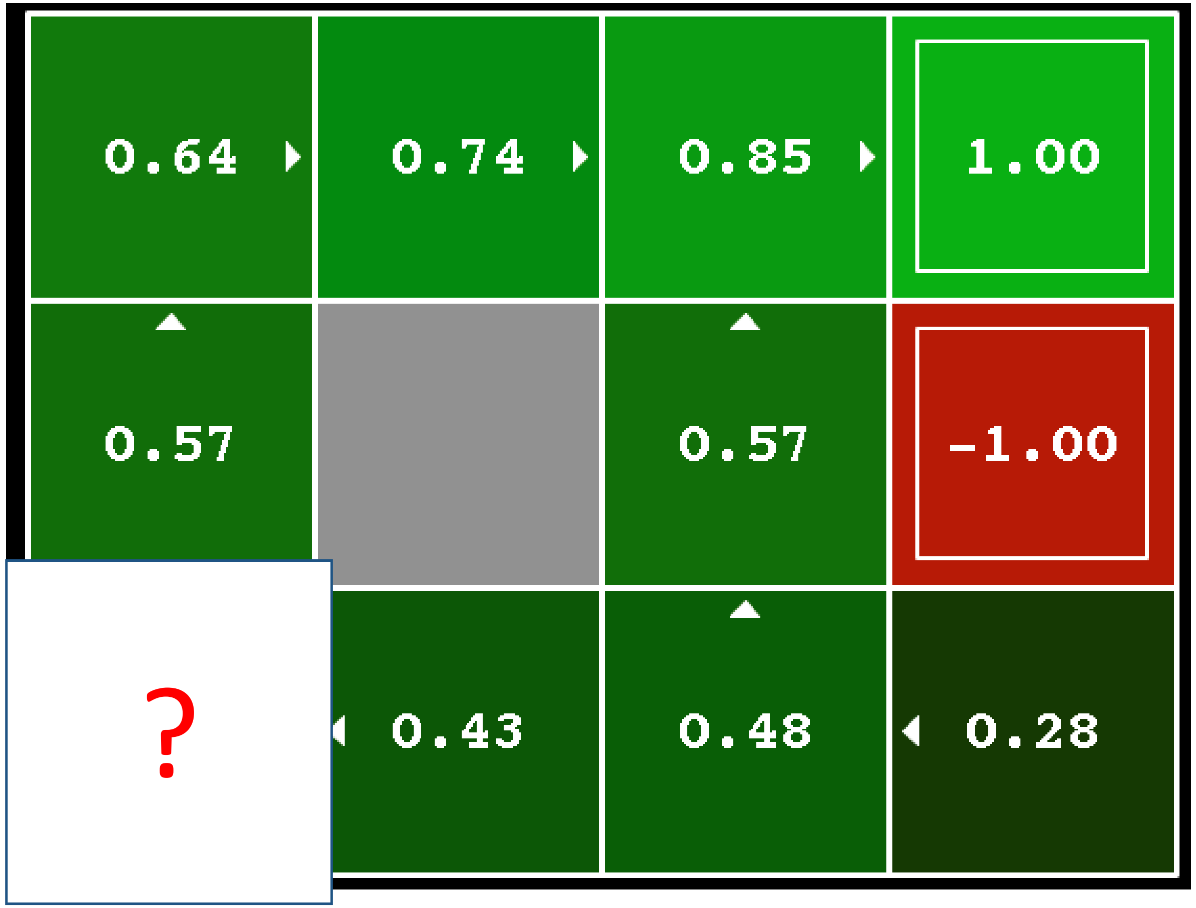
Backup Slides

Snapshot of Demo – Gridworld V Values

What is $V^*(s_{1,1})$?

What is $\pi^*(s_{1,1})$?

V*(s)



Noise = 0.2
Discount = 0.9
Living reward = 0

Snapshot of Demo – Gridworld V Values

What is $V^*(s_{1,1})$?

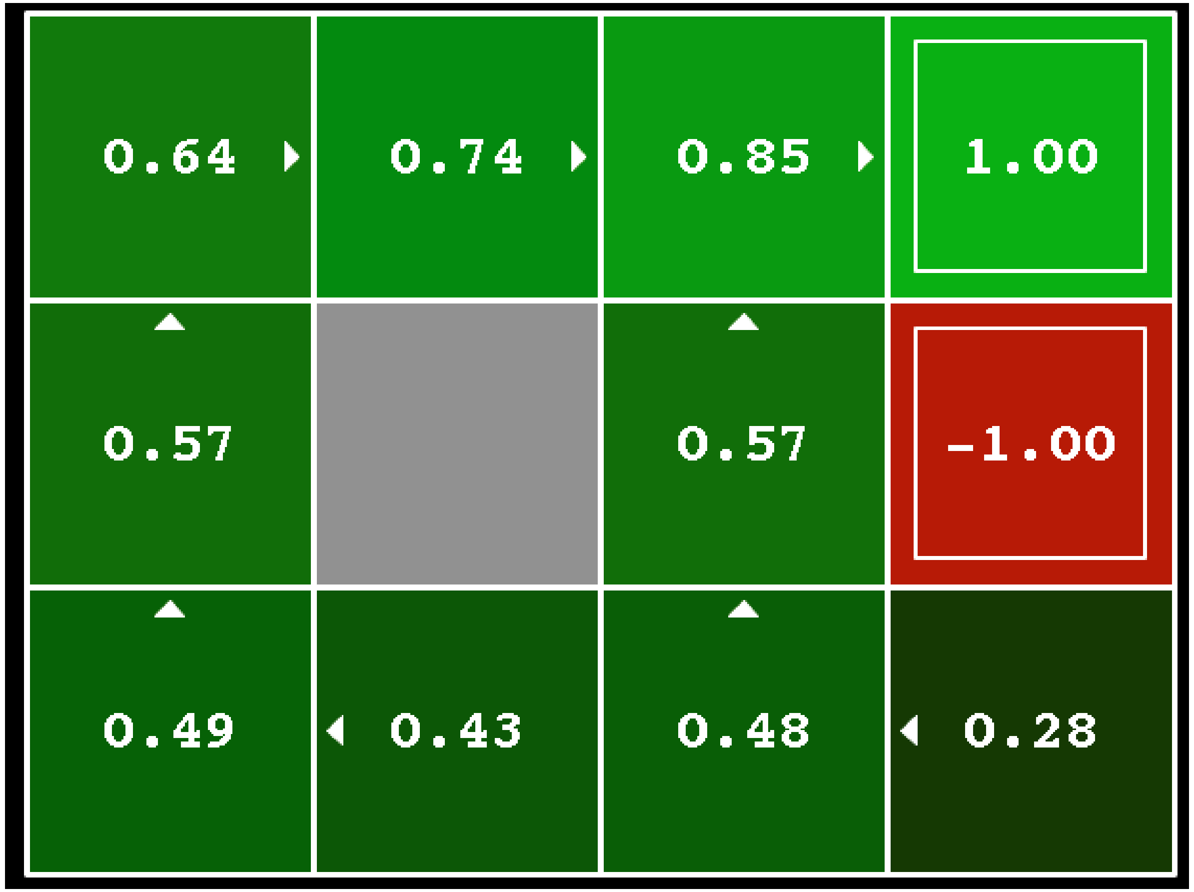
Let $V^*(s_{1,1}) = x$.

If $\pi^*(s_{1,1})$ =North: x=0+0.9*(0.8*0.57+0.1*x+0.1*0.43) ⇒ x=0.493

If $\pi^*(s_{1,1})$ =South: x=0+0.9*(0.8*x+0.1*x+0.1*0.43) ⇒ x=0.203

But when x=0.203, taking action North leads to higher expected value. Contradiction.

What is $\pi^*(s_{1,1})$?



Noise = 0.2
Discount = 0.9
Living reward = 0