Last Time

Last Time

- How is a **Markov decision process** defined? (S, A, T, R, γ)

b initial state dist.

Last Time

- How is a **Markov decision process** defined?
- What is a **policy**?

Last Time

- How is a **Markov decision process** defined?
- What is a **policy**?
- How do we **evaluate** policies?

Guiding Questions

Guiding Questions

- How do we reason about the **future consequences** of actions in an MDP?
- What are the basic **algorithms for solving MDPs**?

Value-Based Policy Evaluation

Want $U(\pi)$

$$U^\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_0 = s\right]$$

$$= E\left[R(s, \pi(s))\right] + \underset{s' \sim T(s,\pi(s))}{E}\left[\sum_{t=1}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_1 = s'\right]$$

$$= R(s, \pi(s)) + \gamma E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_1 = s'\right]$$

$$= R(s, \pi(s)) + \gamma \underset{s' \sim T(s,\pi(s))}{E}\left[U^\pi(s')\right]$$

Bellman Expectation Equation

$$\boxed{U^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s'} T(s' \mid s, \pi(s)) U^\pi(s')}$$

$$\bar{U}^\pi_i = U^\pi(i)$$

$$\bar{T}^\pi_{ij} = T(j \mid i, \pi(i))$$

$$\bar{R}^\pi_i = R(i, \pi(i))$$

$$\bar{U}^\pi = \bar{R}^\pi + \gamma \bar{T}^\pi \bar{U}^\pi$$

$$(I - \gamma \bar{T}^\pi) \bar{U}^\pi = \bar{R}^\pi$$

$$\boxed{\bar{U}^\pi = (I - \gamma \bar{T}^\pi)^{-1} \bar{R}^\pi}$$

$$U(\pi) = \sum_s b(s) U^\pi(s) \qquad \bar{b}_i = b(s)$$

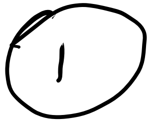
$$\bar{U}^\pi \cdot \bar{b}$$

MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)

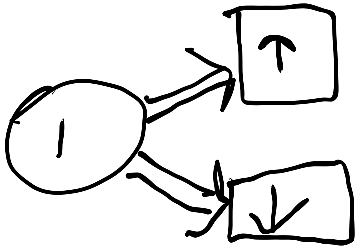
MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)



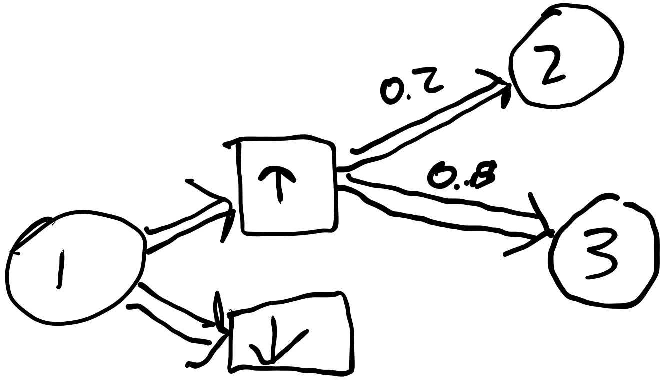
MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)



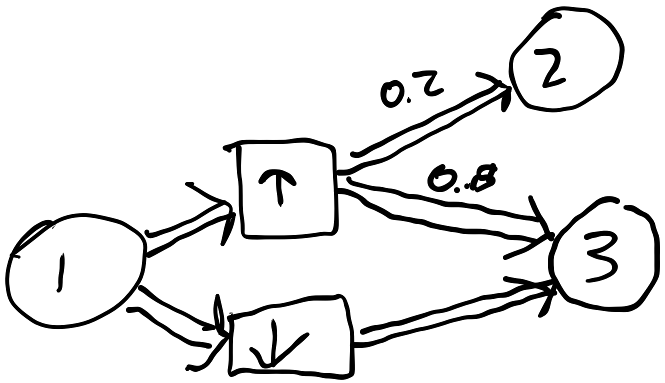
MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)



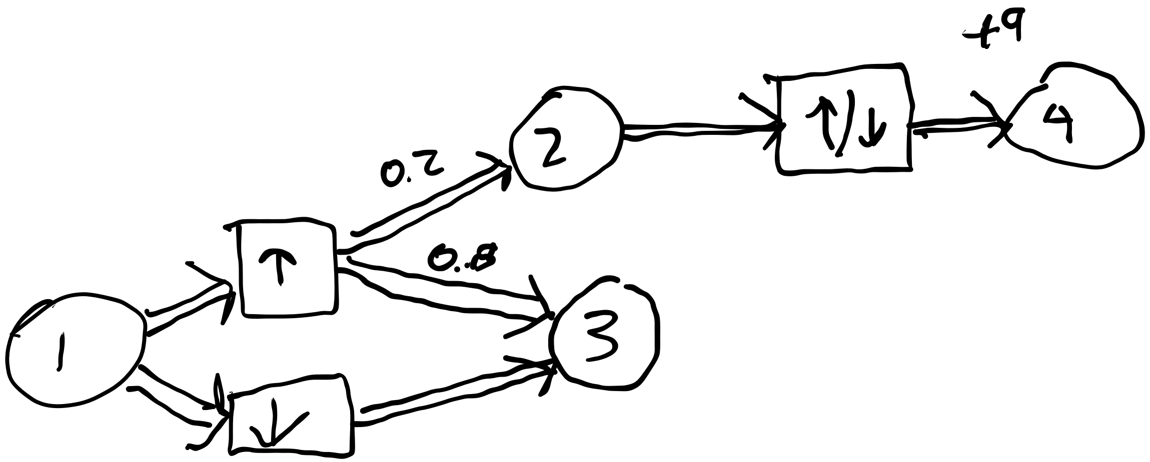
MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)



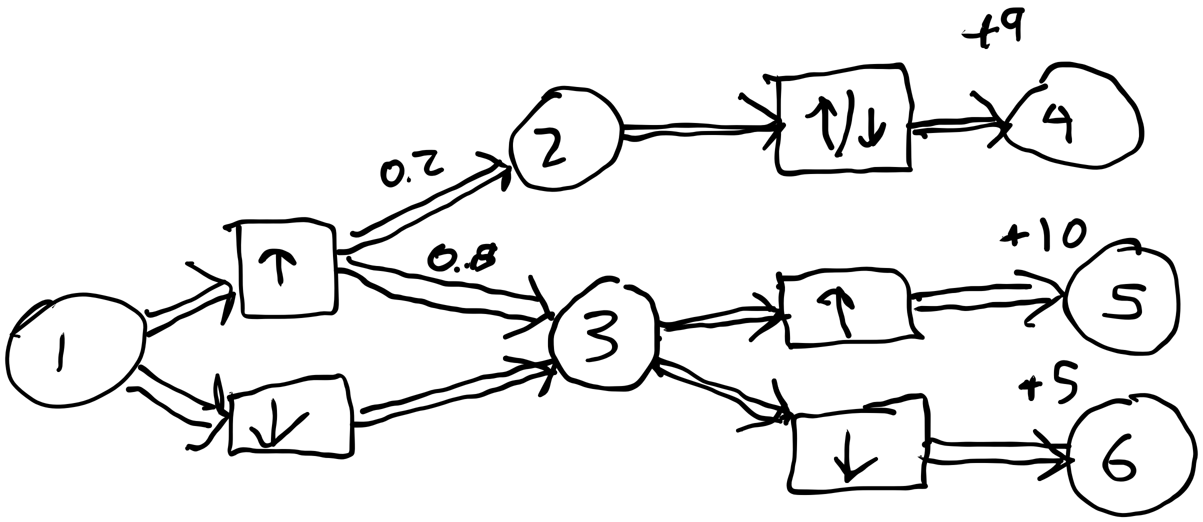
MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)

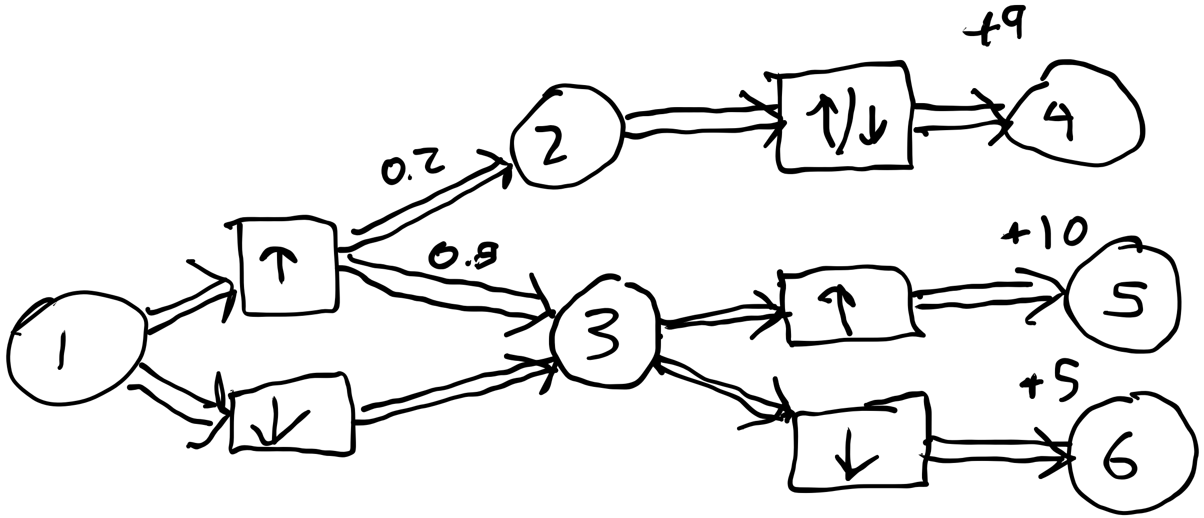


MDP Example: Up-Down Problem

For this lecture, => is same as ->> (distinguishes from Bayes Net)



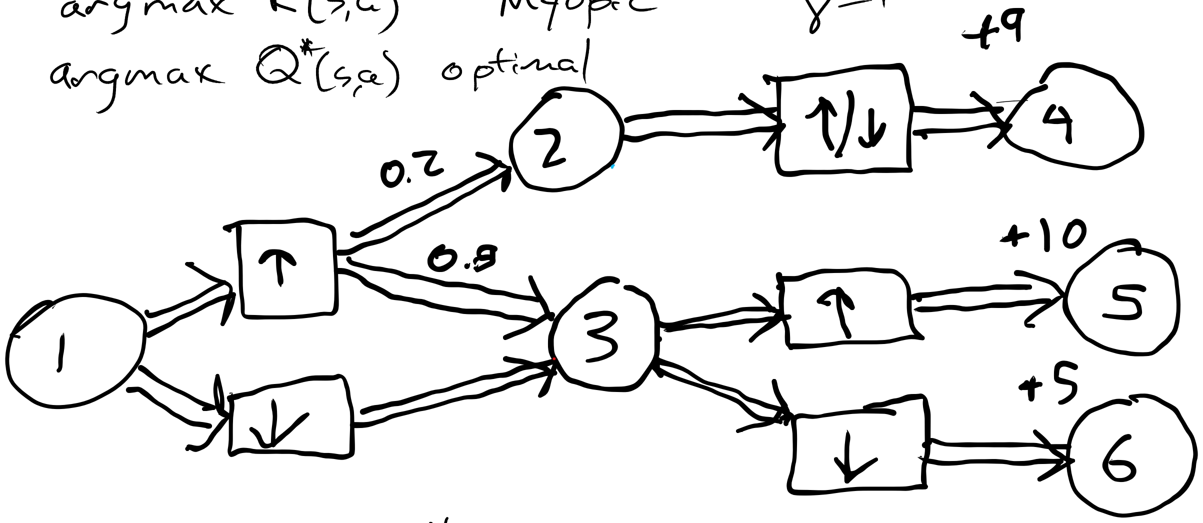
Dynamic Programming and Value Backup



Dynamic Programming and Value Backup

argmax $R(s,a)$ Myopic $\gamma = 1$

argmax $Q^*(s,a)$ optimal



Bellman Backup

$U^*(s) \leftarrow 0$ for all terminal states

Repeat until all $U^*(s)$ calulated

find π^*, U^* for states where U^* is known for all successor states

Only works if there are no cycles?

Bellman's Principle of Optimality: Every sub-policy in an optimal policy is locally optimal

$$U^*(s) = \max_{\pi} U^{\pi}(s)$$ ← optimal value function

$$\pi^*(s) = \underset{a \in A}{\operatorname{argmax}} \underbrace{\left(R(s,a) + \gamma E[U^*(s')] \right)}_{Q^*(s,a)}$$

s	$U^*(s)$	a	$Q^*(s,a)$
4	0		
5	0		
6	0		
2	9	↑/↓	$+9 + \gamma 0$
3	10	↑	$+10 + \gamma 0$
		↓	$+5 + \gamma 0$
1	10	↑	$\underset{=9.8}{0} + \gamma(0.2\, \cancel{U^*(2)} \rightarrow 9 + 0.8\, \cancel{U^*(3)} \rightarrow 10)$
		↓	$\underset{=10}{0} + \gamma(\cancel{U^*(3)} \rightarrow 10)$

Break: DIA Run

Boulder

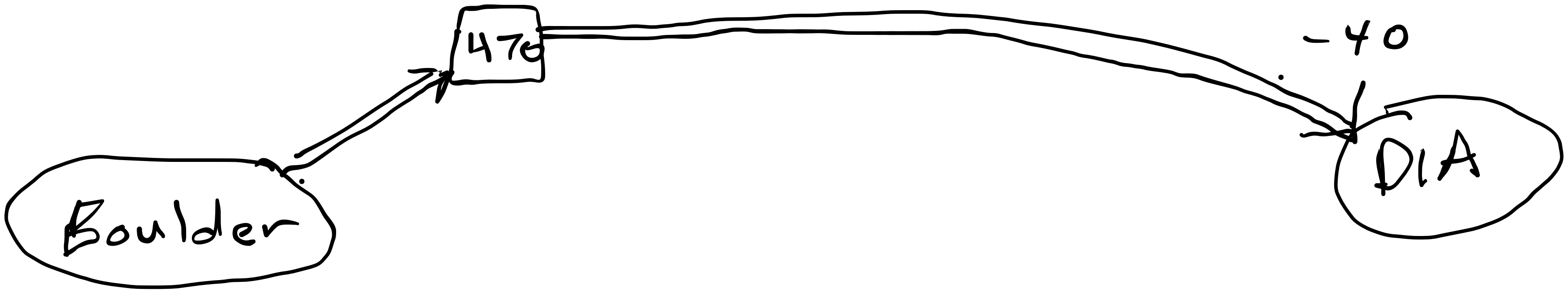
DIA

Break: DIA Run

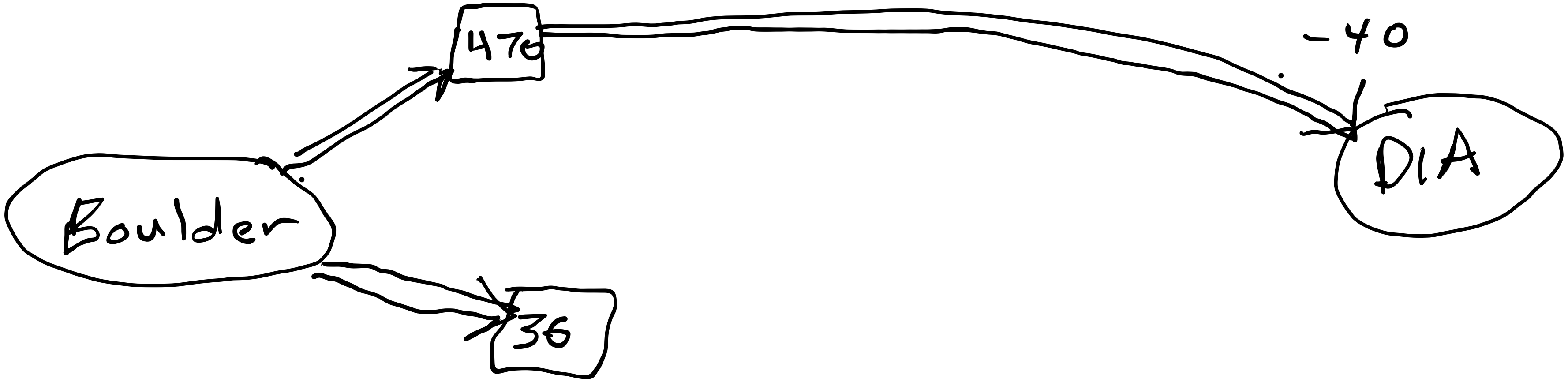
Boulder

DIA

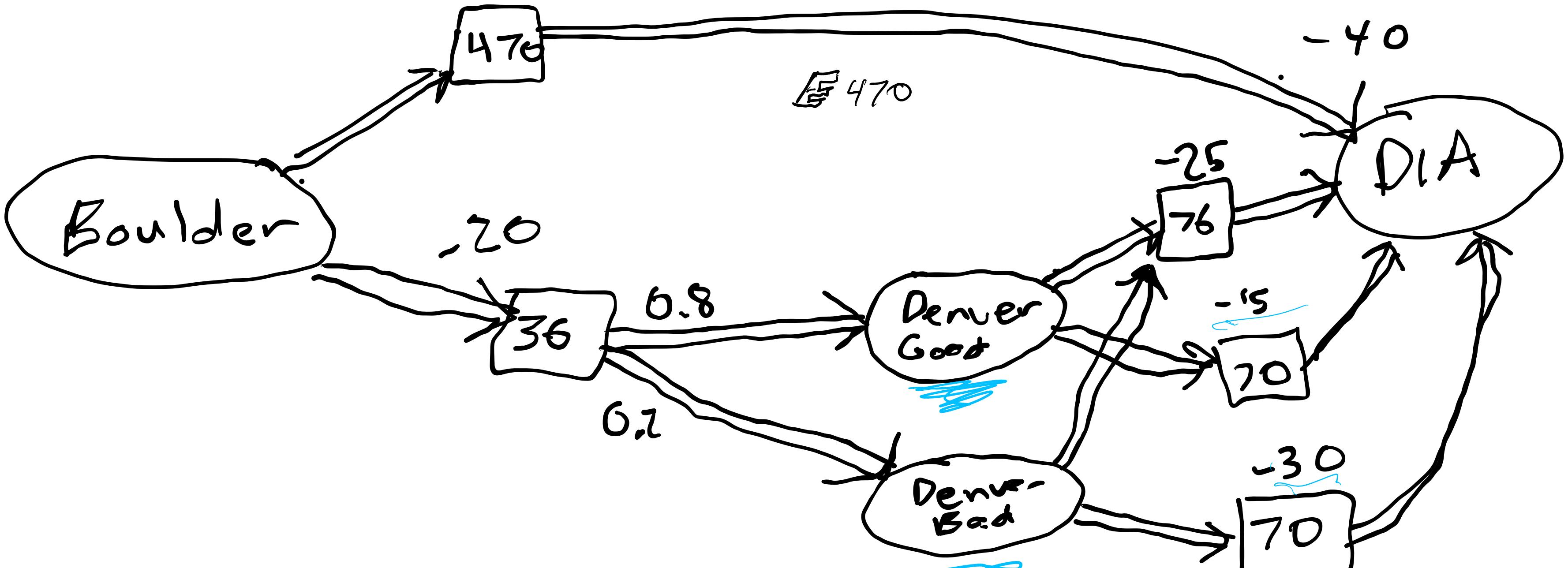
Break: DIA Run



Break: DIA Run



Break: DIA Run



s	$U^*(s)$	a	$Q^*(s,a)$
DIA	0		
Den Good	-15	76	-25
		70	-15
Den Bad	-25	76	-25
		70	-30
Boulder	-37	470	-40
		36	$-20 + 0.2(-25) + 0.8(-15) = -37$

Myopic

$$\operatorname{argmax}_a R(\text{Boulder}, a) = \boxed{36}$$

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

1. initialize π, π' (differently)

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

1. initialize π, π' (differently)
2. while $\pi \neq \pi'$

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

1. initialize π, π' (differently)
2. while $\pi \neq \pi'$
3. $\quad \pi \leftarrow \pi'$

Policy Iteration

U^*

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

$\pi'(s) = \underset{a}{\mathrm{argmax}} \left(R(s,a) + \gamma E\left[U^{\pi}(s') \right] \right)$

1. initialize π, π' (differently)
2. while $\pi \neq \pi'$
3. $\quad \pi \leftarrow \pi'$
4. $\quad U^\pi \leftarrow (I - \gamma T^\pi)^{-1} R^\pi$

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

1. initialize π, π' (differently)
2. while $\pi \neq \pi'$
3. $\quad \pi \leftarrow \pi'$
4. $\quad U^\pi \leftarrow (I - \gamma T^\pi)^{-1} R^\pi$
5. $\quad \pi'(s) \leftarrow \underset{a \in A}{\text{argmax}} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^\pi(s')\right) \quad \forall s \in S$

Policy Iteration

<u>Algorithm: Policy Iteration</u>

Given: MDP (S, A, R, T, γ, b)

1. initialize π, π' (differently)
2. while $\pi \neq \pi'$
3. $\quad \pi \leftarrow \pi'$
4. $\quad U^\pi \leftarrow (I - \gamma T^\pi)^{-1} R^\pi$
5. $\quad \pi'(s) \leftarrow \underset{a \in A}{\text{argmax}} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^\pi(s') \right) \quad \forall s \in S$
6. return π

swap for MC Evaluation

1. Policy Evaluation

2. Policy Improvement

Value Iteration

Algorithm: Value Iteration

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

Value Iteration

Algorithm: Value Iteration

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)

Value Iteration

Algorithm: Value Iteration

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$

Value Iteration

<u>Algorithm: Value Iteration</u>

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$
3. $\quad U \leftarrow U'$

Value Iteration

<u>Algorithm: Value Iteration</u>

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$
3. $\quad U \leftarrow U'$
4. $\quad U'(s) \leftarrow \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^{\cancel{\prime}}(s')\right) \qquad \forall s \in S$

Value Iteration

<u>Algorithm: Value Iteration</u>

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$
3. $\quad U \leftarrow U'$
4. $\quad U'(s) \leftarrow \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^\pi(s')\right) \quad \forall s \in S$
5. return U'

Value Iteration

<u>Algorithm: Value Iteration</u>

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$
3. $\quad U \leftarrow U'$
4. $\quad U'(s) \leftarrow \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^\pi(s')\right) \qquad \forall s \in S$
5. return U'

near

- Returned U' will be U^*!

Value Iteration

<u>Algorithm: Value Iteration</u>

Given: MDP (S, A, R, T, γ, b), tolerance ϵ

1. initialize U, U' (differently)
2. while $\|U - U'\|_\infty < \epsilon$
3. $\quad U \leftarrow U'$
4. $\quad U'(s) \leftarrow \max_{a \in A} \left(\underbrace{R(s,a)}_{\text{Immediate Reward}} + \gamma \underbrace{\sum_{s' \in S} T(s'|s,a) U^\pi(s')}_{\text{Future Value}} \right) \quad \forall s \in S$
5. return U'

- Returned U' will be U^*!
- π^* is easy to extract: $\pi^*(s) = \arg\max(R(s,a) + \gamma E[U^*(s)])$

Bellman's Equations

Policy Evaluation
(Linear)

$$U^{\pi}(s) = R(s,\pi(s)) + \gamma \underset{s' \sim T(s,\pi(s))}{E}\left[U^{\pi}(s')\right]$$

Bellman's Expectation Equation

Bellman Backup
Certificate of Optimality

$$U^{*}(s) = \max_a \left(R(s,a) + \gamma \underset{s' \sim T(s,a)}{E}\left[U^{*}(s')\right]\right)$$

Bellman's Optimality Equation "Bellman's Equation"

Value Iteration

$$U'(s) = B[U](s) = \max_a \left(R(s,a) + \gamma \underset{s' \sim T(s,a)}{E}\left[U(s')\right]\right)$$

Bellman Operator

VI: initialize U, U'
while $\|U - U'\| > \varepsilon$
$U \leftarrow U'$
$U' \leftarrow B[U]$

Guiding Questions

Guiding Questions

- How do we reason about the **future consequences** of actions in an MDP?
- What are the basic **algorithms for solving MDPs**?

$$Q(s,a) = \underbrace{R(s,a)}_{\text{immediate}} + \underbrace{\gamma \underset{s' \sim T(s,a)}{E}\left[U(s')\right]}_{\text{future}}$$

Value Iteration = Repeatedly Applying Bellman Operator

Policy Iteration = Repeatedly
Evaluating Policy
Improving Policy