Last Time

Last Time

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

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**?

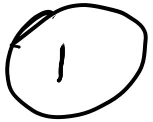
Value-Based Policy Evaluation

MDP Example: Up-Down Problem

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

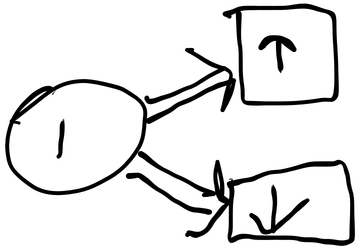
MDP Example: Up-Down Problem

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



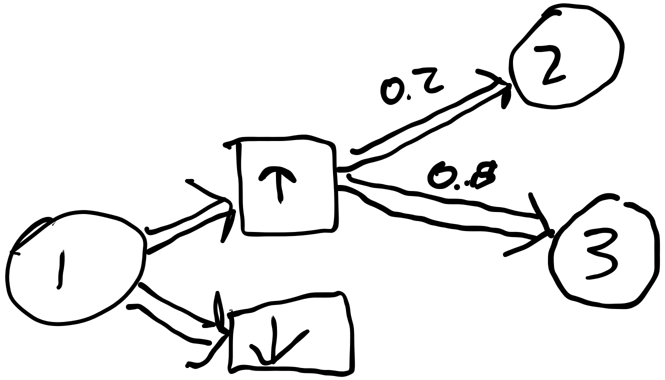
MDP Example: Up-Down Problem

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



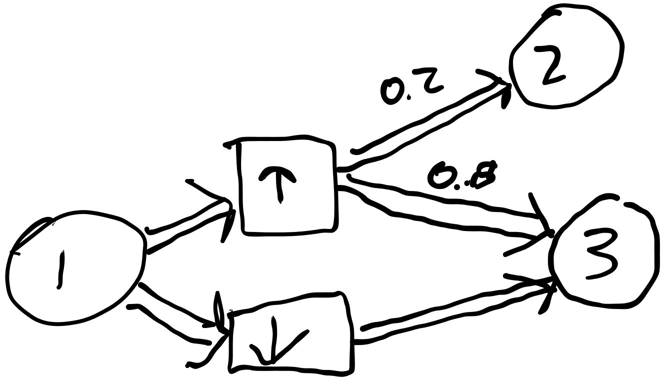
MDP Example: Up-Down Problem

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



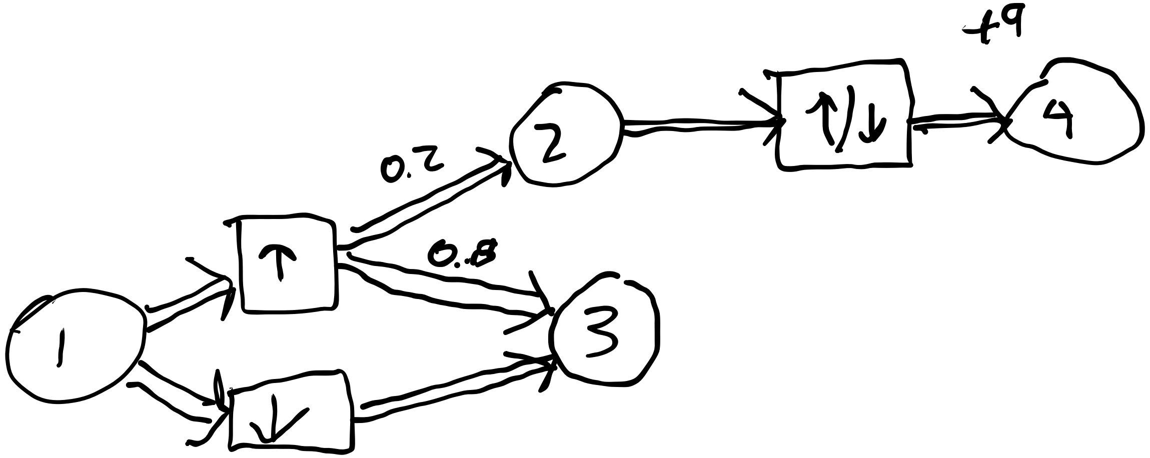
MDP Example: Up-Down Problem

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



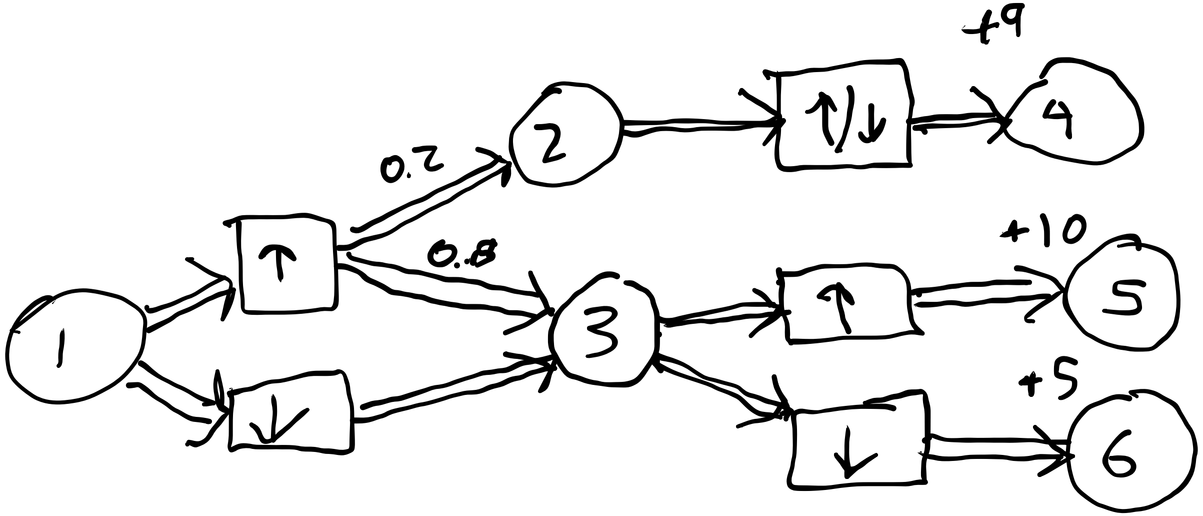
MDP Example: Up-Down Problem

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

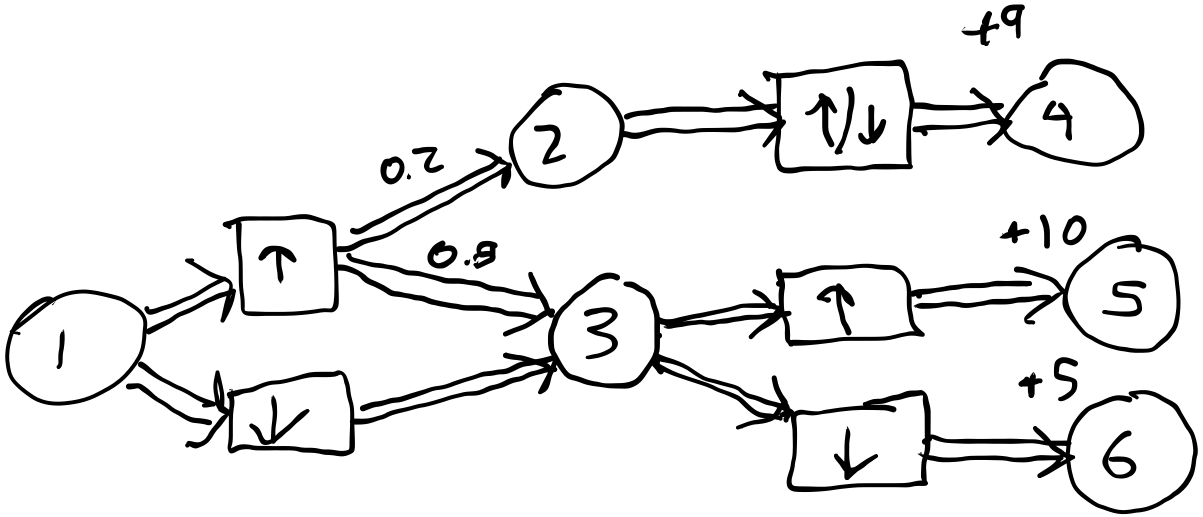


MDP Example: Up-Down Problem

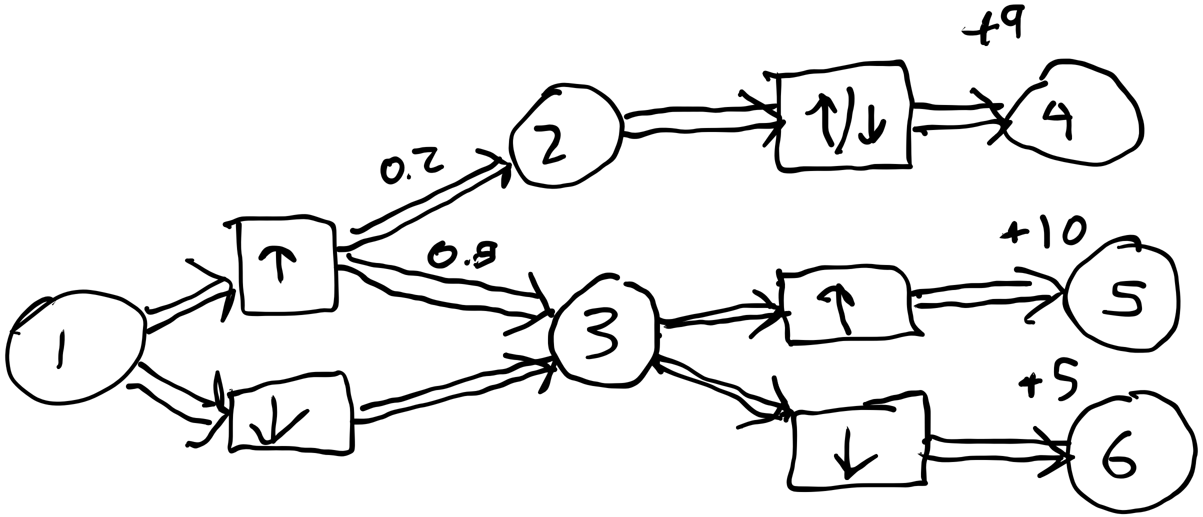
For this lecture, => is same as ->> (distinguishes from Bayes Net)



Dynamic Programming and Value Backup



Dynamic Programming and Value Backup

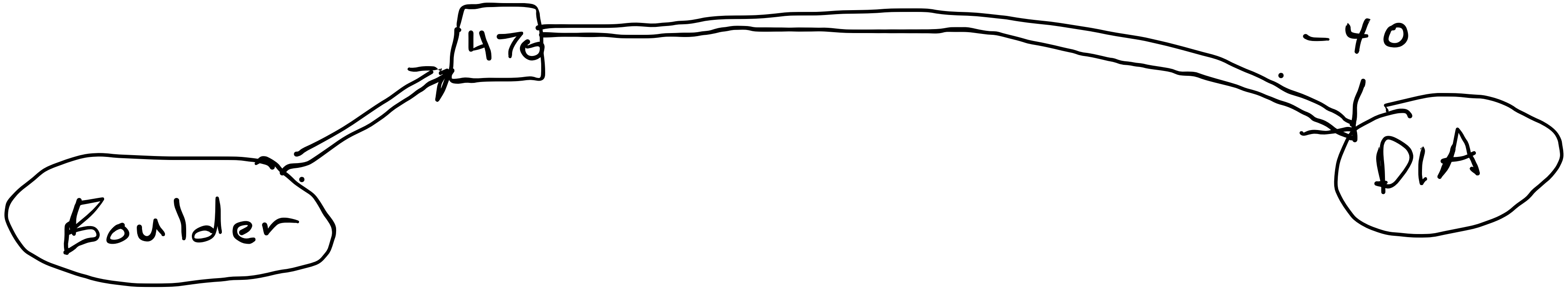


Bellman's Principle of Optimality: Every sub-policy in an optimal policy is locally optimal

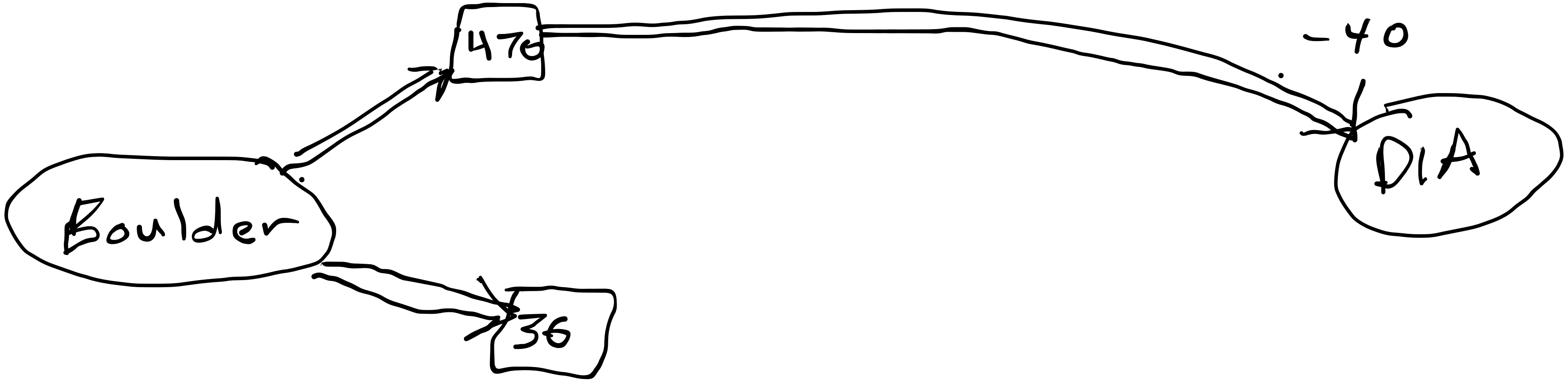
Break: DIA Run

Break: DIA Run

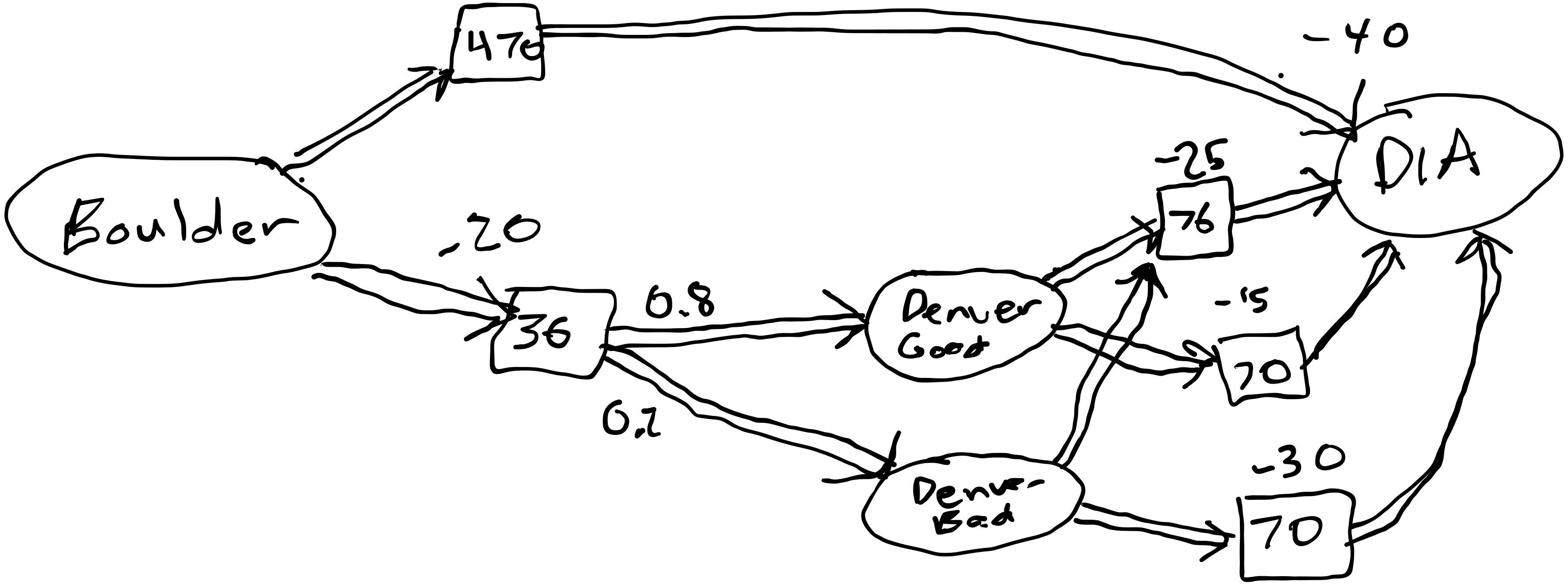
Break: DIA Run



Break: DIA Run



Break: DIA Run



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 $\pi$, $\pi'$ (differently)
```

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

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

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

```
initialize $\pi$, $\pi'$ (differently)
while $\pi \neq \pi'$
    $\pi \leftarrow \pi'$
```

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

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

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

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

Policy Iteration

Algorithm: Policy Iteration

Given: MDP (S, A, R, T, γ, b)

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

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

Algorithm: Value Iteration

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

```
initialize $U$, $U'$ (differently)
while $\|U - U'\|_\infty < \epsilon$
    $U \leftarrow U'$
```

Value Iteration

Algorithm: Value Iteration

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

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

Value Iteration

Algorithm: Value Iteration

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

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

Value Iteration

Algorithm: Value Iteration

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

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

- 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.     $U \leftarrow U'$
4.     $U'(s) \leftarrow \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) U^\pi(s')\right) \quad \forall s \in S$
5. return $U'$
```

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

Bellman's Equations

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**?