

MDPs

MDPs Calculating Pol

Solving MDPs

Markov Decision Processes

School of Computing and Data Science



MDPs

MDPs MDPs

Calculating Polici

Solving MDPs

- **discounted reward**: penalize future rewards by γ
 - $R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots + \gamma^n R(s_n)$
- **policy**: $\pi(s) = a$ gives an action for each state
- optimal policy: π^*



Calculating π^*

MDPs Calculating Policies

Solving MDPs

Optimal policy is based on optimal utility:

$$\pi^*(s) = \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} T(s, a, s') U^{\pi^*}(s')$$

Utility of a policy is based on expected rewards:

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

Utility of a state is its reward plus the best utility of an action in that state:

School of Computing and Data Science



Calculating π^*

Optimal policy is based on optimal utility:

$$\pi^*(s) = \operatorname*{argmax}_{a \in A(s)} \sum_{s'} T(s, a, s') U^{\pi^*}(s')$$

Utility of a policy is based on expected rewards:

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

Utility of a state is its reward plus the best utility of an action in that state:

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s, a, s') U(s')$$

School of Computing and Data Science



Calculating π^*

Optimal policy is based on optimal utility:

$$\pi^*(s) = \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} T(s, a, s') U^{\pi^*}(s')$$

Utility of a policy is based on expected rewards:

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

Utility of a state is its reward plus the best utility of an action in that state:

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s, a, s') U(s')$$

School of Computing and Data Science



MDP

Solving MDPs

Value Iteration

Solving MDPs

School of Computing and Data Science



Value Iteration

MDPs

Solving MDPs

Value Iteration

Example

Repeated Bellman updates:

- 1: for all States *s* do
- 2: $U(s) \leftarrow R(s)$
- 3: while unsatisfied do
- 4: **for each** state *s* **do**

5:
$$U'(s) \leftarrow R(s) + \gamma \max_{a \sum_{s'} T(s, a, s')} U(s')$$

6: $U \leftarrow U'$

Utility values are guaranteed to converge with enough updates This equilibrium gives an optimal policy



Transitions to terminal states have rewards of -1 and 1, all other transition rewards are -.04

Probability of .8 to move in intended direction, .1 to move at a right angle

$$0 < \gamma \le 1$$
 – let's pick $\gamma = .5$

School of Computing and Data Science

0.1