Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 3

Sep, 9 2019

Lecture Overview

Markov Decision Processes

- Formal Specification and example
- Policies and Optimal Policy
- Intro to Value Iteration

Combining ideas for Stochastic planning

- What is a key limitation of decision networks?

Represent (and optimize) only a fixed number of decisions

- What is an advantage of Markov models?

The network can extend indefinitely

Goal: represent (and optimize) an indefinite sequence of decisions

Decision Processes

Often an agent needs to go beyond a fixed set of decisions – Examples?

- Would like to have an **ongoing decision process**

Infinite horizon problems: process does not stop

Robot surviving on planet, Monitoring Nuc. Plant, ...

Indefinite horizon problem: the agent does not know when the process may stop

reaching location

Finite horizon: the process must end at a give time N

in N steps

Markov Models

Markov Chains

Noisy
observations

Hidden Markov Model

Partially Observable Markov Decision Processes (POMDPs)

Noisy
Actions
Rewards

Markov Decision Processes (MDPs)

How can we deal with indefinite/infinite Decision processes?

We make the same two assumptions we made for….

The action outcome depends only on the current state

Markov

Let S_t be the state at time t …

$P(S_{t+1} \mid S_t, A_t, S_{t-1}, A_{t-1} \ldots)$

$P(S_{t+1} \mid S_t, A_t)$

The process is *stationary*…

the same for all t

We also need a more flexible specification for the utility. How?

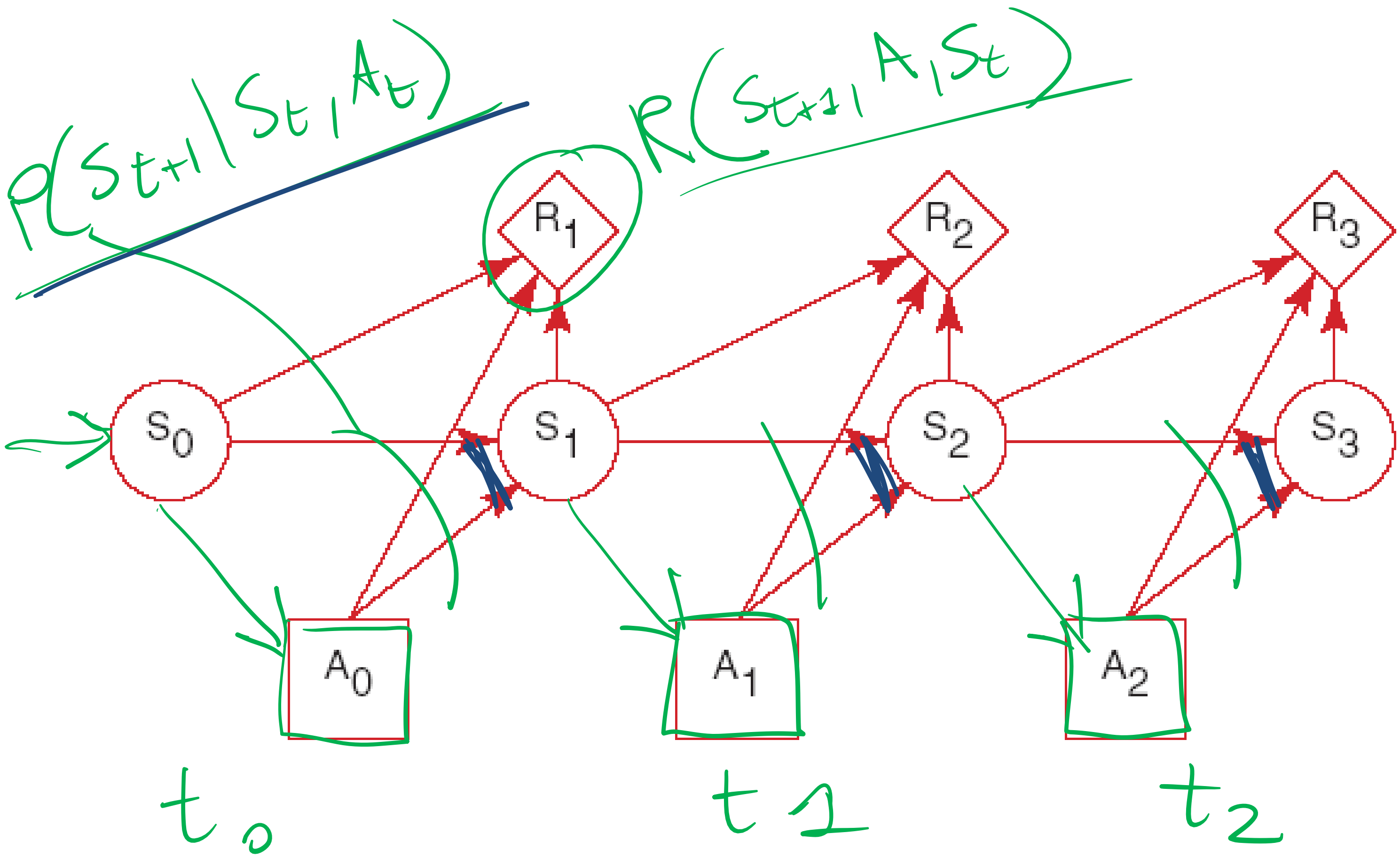
- Defined based on a reward/punishment *R(s)* that the agent receives in each state *s*

$s_0 \quad s_1 \ldots\ldots\ldots s_n$

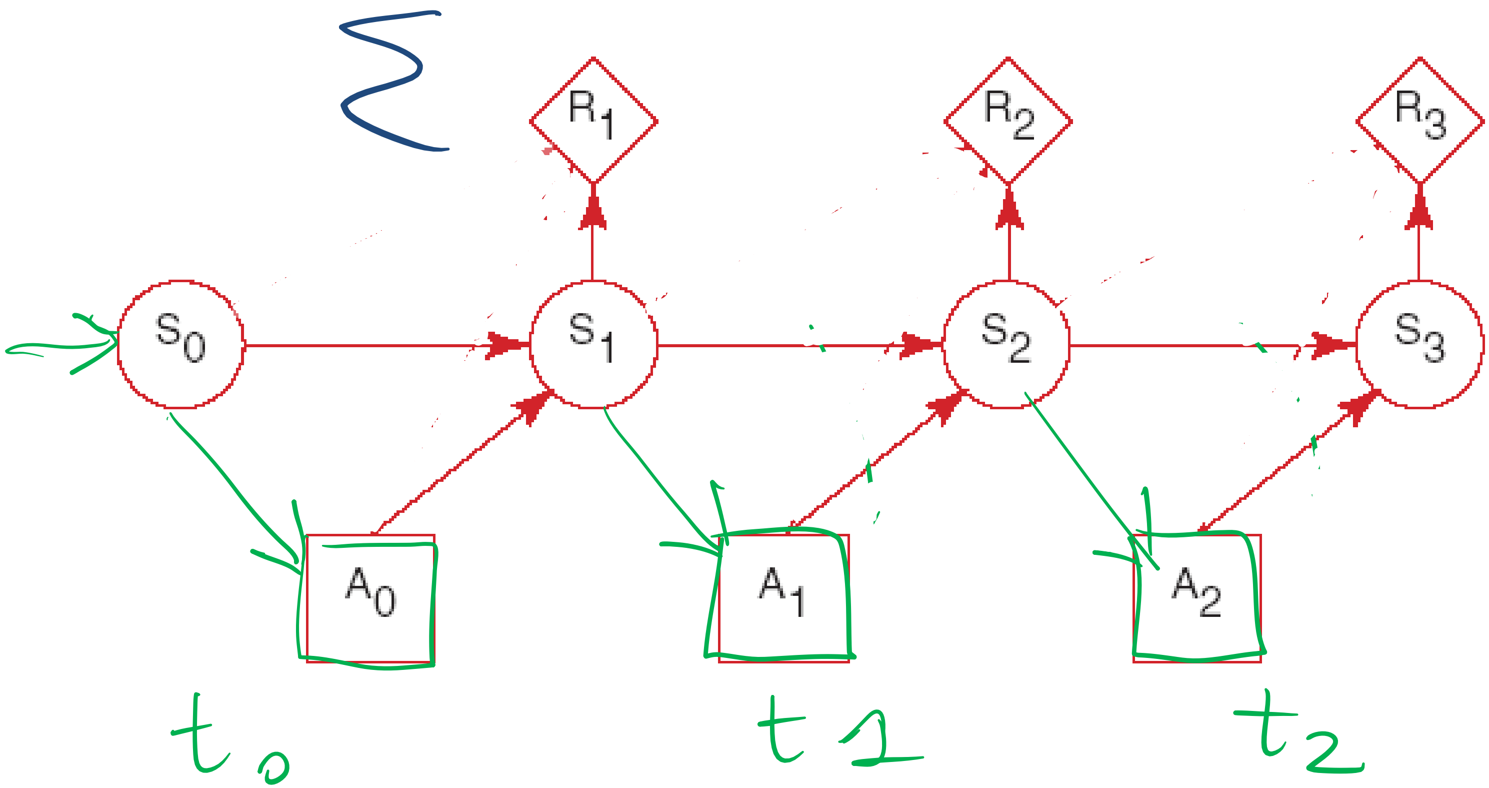
e.g. $\sum \; r_0 \quad r_1 \ldots\ldots\ldots r_n$

MDP graphical specification

Basically a MDP is a Markov Chain augmented with actions and rewards/values



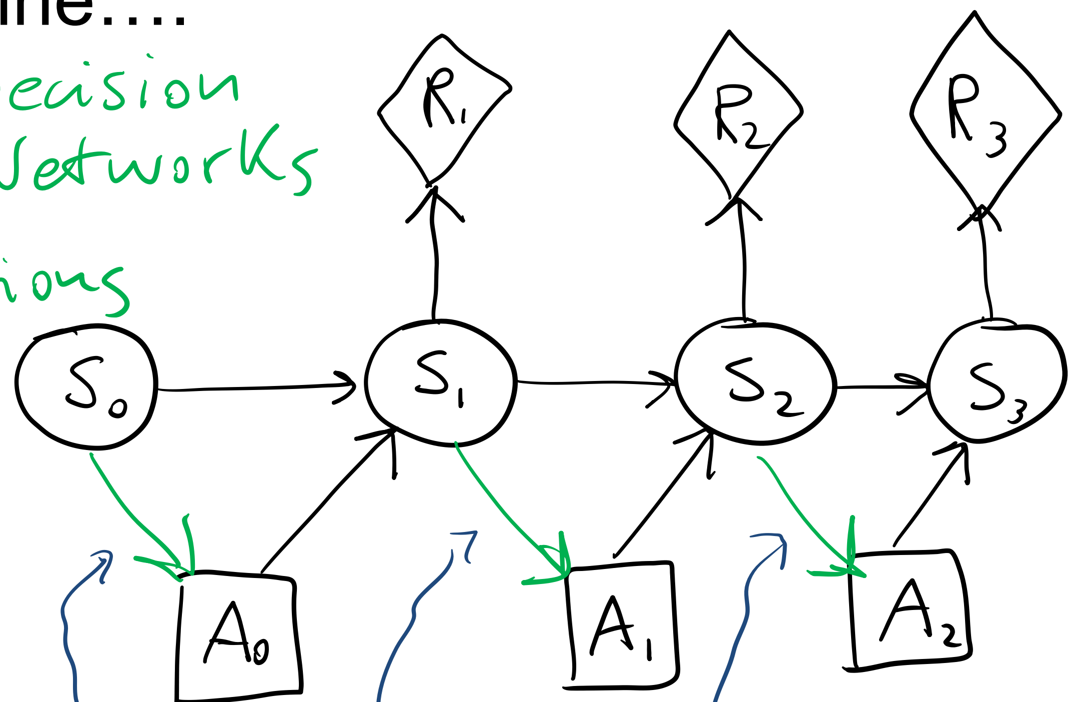
When Rewards only depend on the state



Summary Decision Processes: MDPs

To manage an ongoing (indefinite… infinite) decision process, we combine….

Markov Chains & Decision Networks



Markovian

Stationary

Assumptions

Utility not just at the end

BUT

Sequence of rewards

Fully Observable

MDP: formal specification

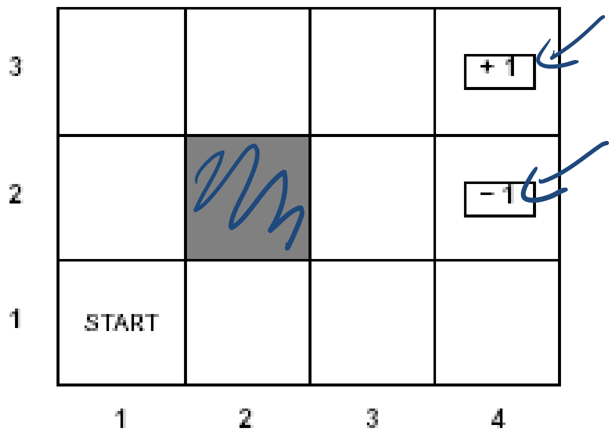
For an MDP you specify:

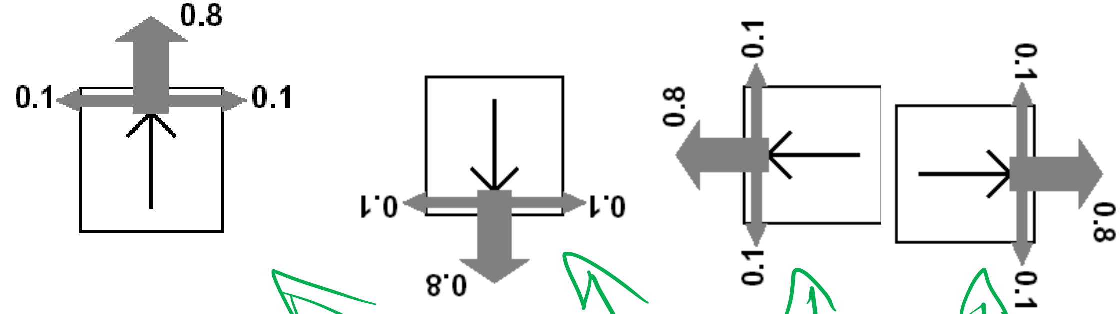
- set S of states and set A of actions
- the process' dynamics (or ***transition model***)

  $P(S_{t+1}|S_t, A_t)$
- The **reward function**
  - $R(s)$ is used when the reward depends only on the state **s** and not on how the agent got there
  - More complex $R(s, a, s')$ describing the reward that the agent receives when it performs action a in state s and ends up in state s'
- **Absorbing/stopping/terminal state** S_{ab}

for all action $P(S_{ab} | a, S_{ab}) = 1$ $R(S_{ab}, a, S_{ab}) = 0$

Example MDP: Scenario and Actions





Agent moves in the above grid via actions *Up, Down, Left, Right*

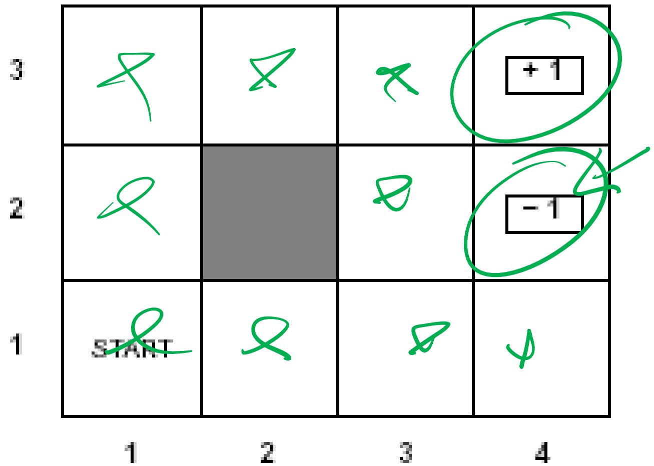
Each action has:

- 0.8 probability to reach its intended effect
- 0.1 probability to move at right angles of the intended direction
- If the agents bumps into a wall, it says there

How many states? = {(11),(12), ... ,(24),(34)}

There are two terminal states (3,4) and (2,4)

Example MDP: Rewards

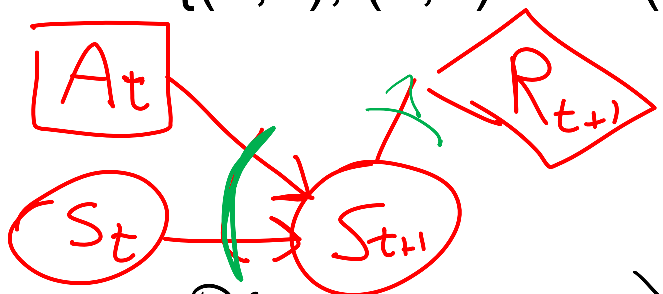


$$R(s) = \begin{cases} -0.04 & \text{(small penalty) for nonterminal states} \\ \pm 1 & \text{for terminal states} \end{cases}$$

Example MDP: Underlying info structures

Four actions *Up, Down, Left, Right*

Eleven States: {(1,1), (1,2)…… (3,4)}



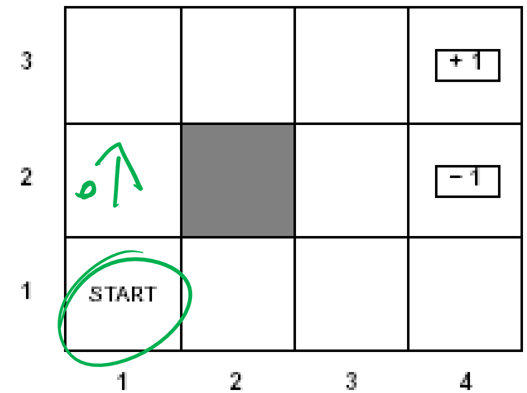


Table $4 \times 11 \times 11$ $P(S_{t+1} \mid S_t, A_t)$

Up

	1,1	2,1	1,2	3,1
1,1	.1	.8	.1	0 0 0 0 0
2,1	0	.2	0	.8
⋮				

Down

L

R

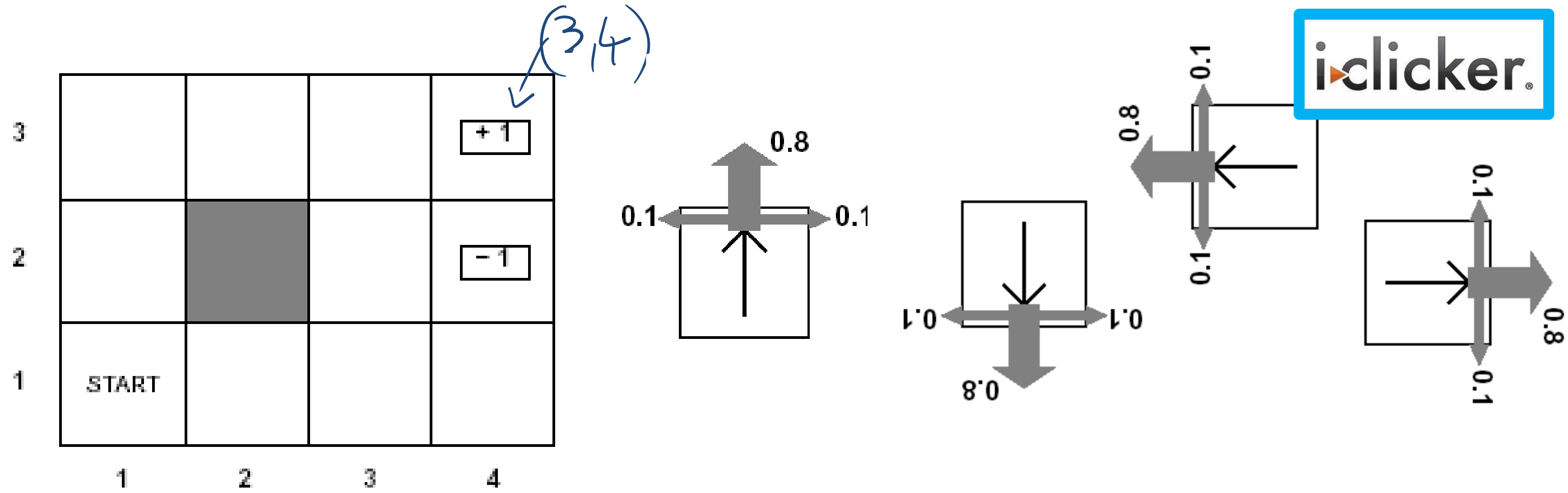
$P(S_0)$

1,1	1
⋮	0
	0
	0
	0
	0
	0

$R(s)$

11	−.04
⋮	⋮
(2 4)	−1
(3 4)	+1

Example MDP: Sequence of actions



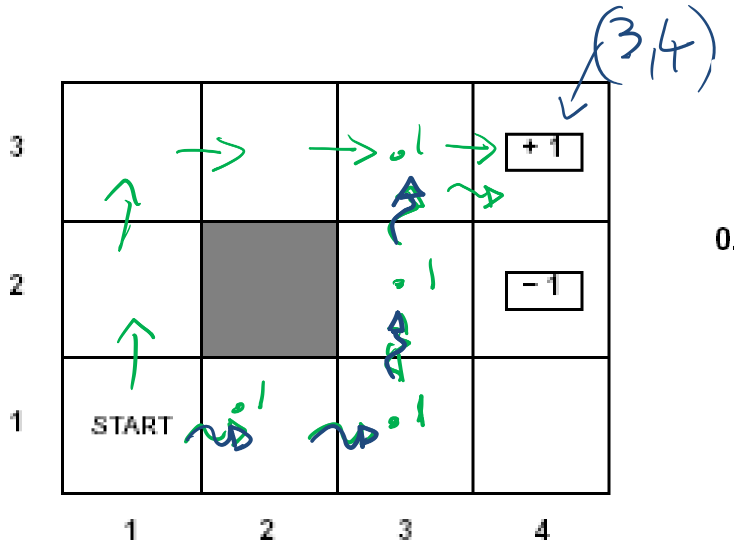
The sequence of actions [*Up, Up, Right, Right, Right*] will take the agent in terminal state (3,4)…

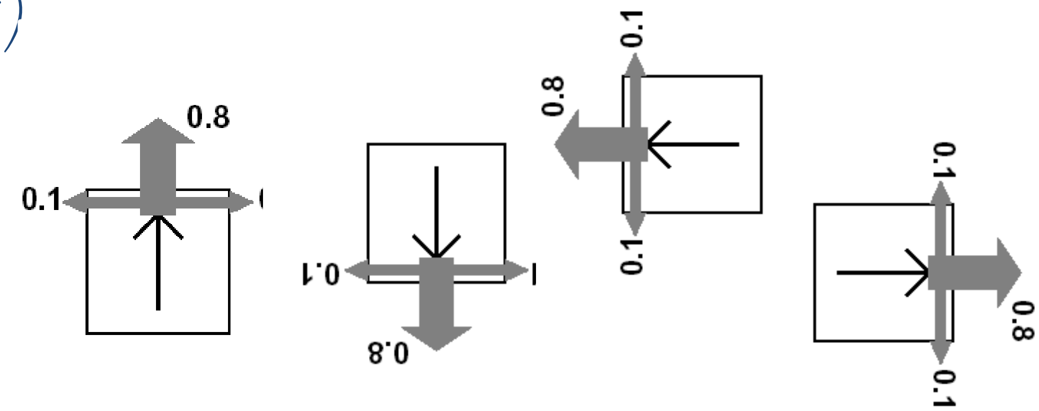
A. ***always*** **B.** ***never*** **C.** ***Only sometimes***

With what probability?

A. **$(0.8)^5$** **B.** **$(0.8)^5 + ((0.1)^4 \times 0.8)$** **C.** **$((0.1)^4 \times 0.8)$**

Example MDP: Sequence of actions





Can the sequence [*Up, Up, Right, Right, Right*] take the agent in terminal state (3,4)?

$(.8)^5$

Can the sequence reach the goal in any other way?

$(.1)^4 \ .8$ ← with prob

yes

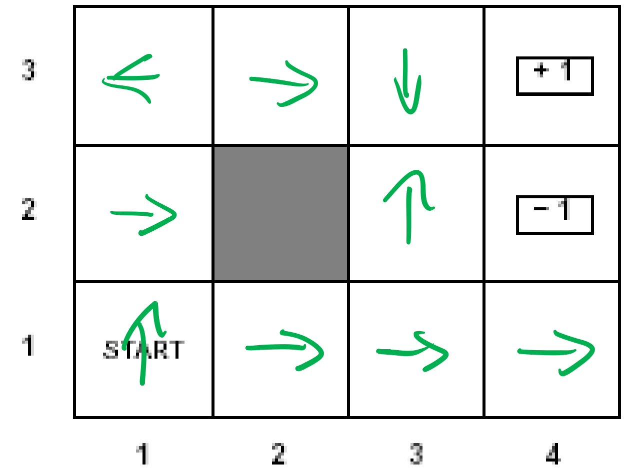
MDPs: Policy

- The robot needs to know what to do as the decision process unfolds…
- It starts in a state, selects an action, ends up in another state selects another action….

- Needs to make the same decision over and over: Given the current state what should I do?

policy

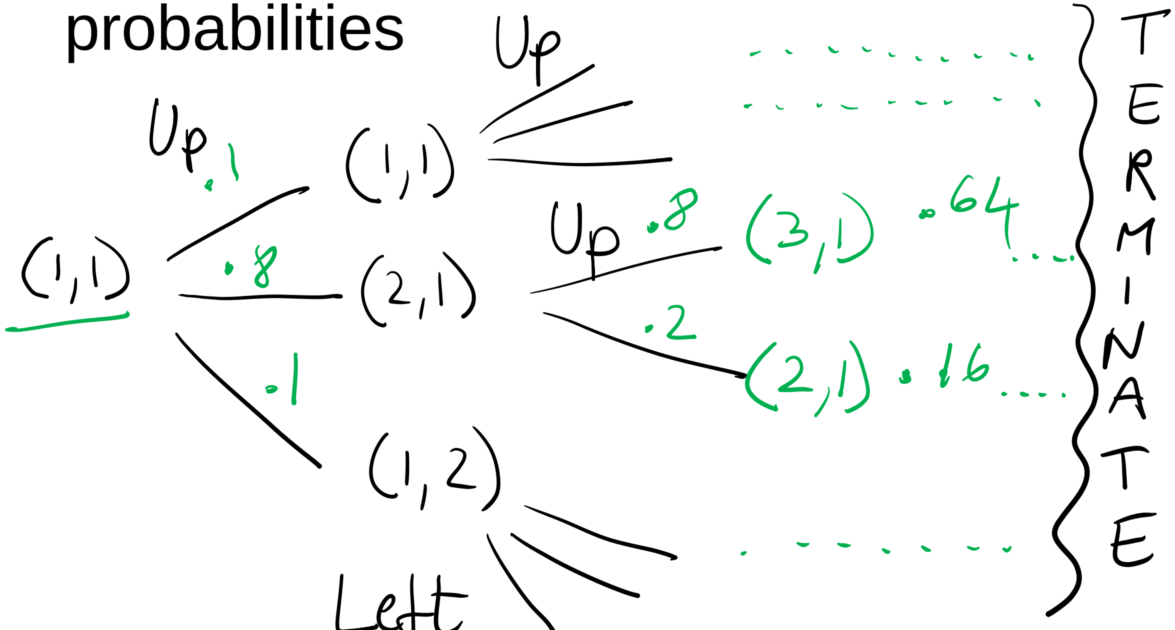
- So a policy for an MDP is a single decision function *π(s)* that specifies what the agent should do for each state *s*

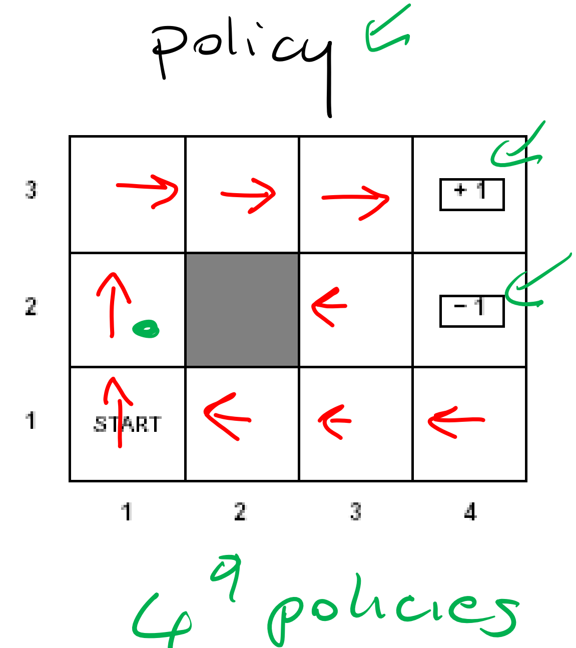


How to evaluate a policy

(in essence how to compute $V^{\pi}(s)$ brute force)

A policy can generate a set of state sequences with different probabilities





Each state sequence has a corresponding reward. Typically the (*discounted*) sum of the rewards for each state in the sequence

$$\sum \; (1,1) \xrightarrow{-.04} (1,1) \xrightarrow{-.04} (2,1) \rightarrow (3,1) \rightarrow (3,1) \rightarrow (3,2) \rightarrow (3,3) \rightarrow (3,4) \;\; +1$$

$+.72$

MDPs: expected value/total reward of a policy and optimal policy

Each sequence of states (environment history) associated with a policy has

- a certain **probability** of occurring
- a given amount of total **reward** as a function of the rewards of its individual states

Expected value /total reward

$$\sum \underbrace{P(s_0, s_1, \ldots, s_{TERMINAL})}_{\text{probability}} * \underbrace{\sum R(s_0) \ldots R(s_T)}_{\text{rewards}}$$

For all the sequences of states generated by the policy

we sum the product of its probability times its reward

Optimal policy is the policy that maximizes *expected total reward*

Lecture Overview

Markov Decision Processes

- Formal Specification and example
- Policies and Optimal Policy
- **Intro to Value Iteration**

Sketch of ideas to find the optimal policy for a MDP (Value Iteration)

We first need a couple of definitions

- **V^π(s)**: the expected value of following policy π in state s
- **Q^π(s, a)**, where a is an action: expected value of performing *a* in *s*, and then following policy π.

Can we express **Q^π(s, a)** in terms of **V^π(s)** ?

$$Q^\pi(s, a) = V^\pi(s) + R(s)$$

A.

$$Q^\pi(s, a) = R(s) + \sum_{s' \in X} P(s' | s, a) * V^\pi(s')$$

B.

$$Q^\pi(s, a) = R(s) + \sum_{s' \in X} V^\pi(s')$$

C.

D. None of the above

X: set of states reachable from *s* by doing *a*

Discounted Reward Function

- Suppose the agent goes through states $s_1, s_2, ..., s_k$ and receives rewards $r_1, r_2, ..., r_k$
- We will look at ***discounted reward*** to define the reward for this sequence, i.e. its *utility* for the agent

$$\gamma \;\; \textbf{\textit{discount factor}},\; 0 \le \gamma \le 1$$

$$U[s_1, s_2, s_3, ..] = r_1 + \gamma r_2 + \gamma^2 r_3 + ...$$

Sketch of ideas to find the optimal policy for a MDP (Value Iteration)

We first need a couple of definitions

- $V^{\pi}(s)$: the expected value of following policy π in state s
- $Q^{\pi}(s, a)$, where *a* is an action: expected value of performing *a* in *s*, and then following policy π.

We have, by definition

$$Q^{\pi}(s, a) = R(s) + \gamma \sum_{s'} P(s' \mid s, a) V^{\pi}(s')$$

reward obtained in *s*

Discount factor

states reachable from *s* by doing *a*

Probability of getting to *s'* from *s* via *a*

expected value of following policy π in *s'*

Value of a policy and Optimal policy

We can also compute *V ^π^(s)* in terms of *Q ^π^(s, a)*

$$V^{\pi}(s) = Q^{\pi}(s, \pi(s))$$

Expected value of following π in s

Expected value of performing the action indicated by π in s and following π after that

action indicated by π in s

For the optimal policy π^* we also have

$$V^{\pi^*}(s) = Q^{\pi^*}(s, \pi^*(s))$$

Value of Optimal policy

$$V^{\pi^*}(s) = Q^{\pi^*}(s, \pi^*(s))$$

Remember for any policy π

$$Q^{\pi}(s, \pi(s)) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s)) \times V^{\pi}(s'))$$

But the Optimal policy π^* is the one that gives the action that maximizes *the future reward* for each state

$$\overset{V^{\pi^*}(s)}{\cancel{Q^{\pi^*}(s, \pi^*(s))}} = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a) \times V^{\pi^*}(s')$$

So…

$$V^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a) \times V^{\pi^*}(s'))$$

Value Iteration Rationale

- Given *N* states, we can write an equation like the one below for each of them

$$\boxed{V(s_1) = R(s_1) + \gamma \max_a \sum_{s'} P(s'|s_1,a)V(s')}$$

$$\boxed{V(s_2) = R(s_2) + \gamma \max_a \sum_{s'} P(s'|s_2,a)V(s')}$$

- Each equation contains *N* unknowns – the V values for the *N* states
- N equations in N variables (Bellman equations): It can be shown that they have a unique solution: the values for the optimal policy
- Unfortunately the N equations are non-linear, because of the max operator: Cannot be easily solved by using techniques from linear algebra
- **Value Iteration Algorithm**: Iterative approach to find the optimal policy and corresponding values

Learning Goals for today's class

You can:

- Compute the probability distribution on states given a sequence of actions in an MDP
- Define a policy for an MDP
- Define and Justify a discounted reward function
- Derive the Bellman equations on which Value Iteration is based (we will likely finish this in the next lecture)

TODO for Wed

Read textbook

- **9.5.3 Value Iteration**

CPSC 322 Review “Exam”

https://forms.gle/SpQwrXfonTZrVf4P7

Based on CPSC 322 material

- Logic
- Uncertainty
- Decision Theory

Review material (e.g., 322 slides from 2017):

https://www.cs.ubc.ca/~carenini/TEACHING/CPSC322-17S/index.html