Department of Computer Science
Undergraduate Events

More details @ https://my.cs.ubc.ca/students/development/events

Deloitte Info Session

Mon., Sept 14
6 – 7 pm
DMP 310

Google Info Table

Mon., Sept 14
10 – 11:30 am; 2 – 4 pm
Reboot Cafe

Google Alkumni/Intern Panel

Tues., Sept 15
6 – 7:30 pm
DMP 310

Co-op Info Session

Thurs., Sept 17
12:30 – 1:30 pm
MCLD 202

Simba Technologies Tech Talk/Info Session

Mon., Sept 21
6 – 7 pm
DMP 310

EA Info Session

Tues., Sept 22
6 – 7 pm
DMP 310

Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 3

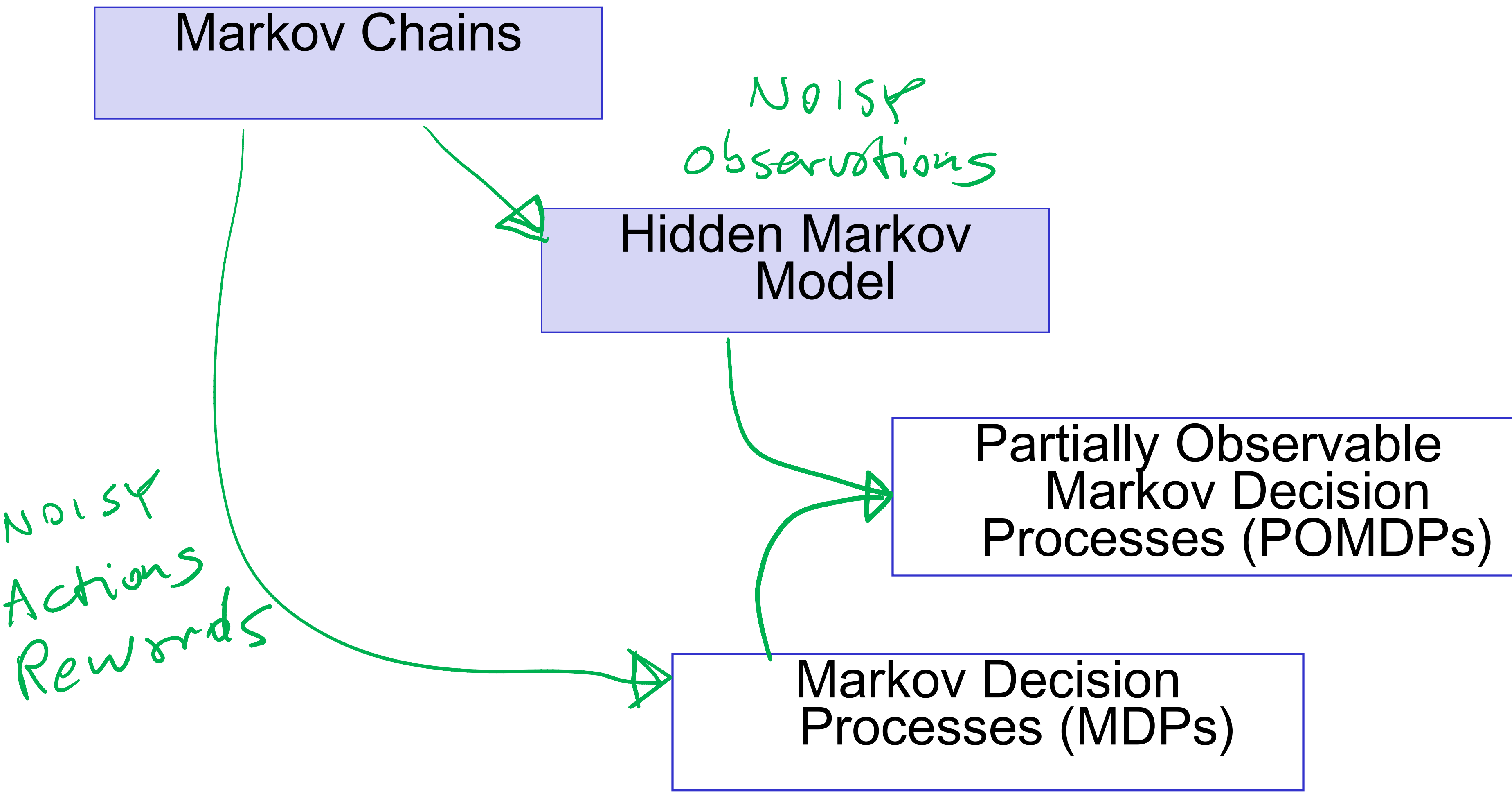
Sep, 14, 2015

Lecture Overview

Markov Decision Processes

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

Markov Models



Summary Decision Processes: MDPs

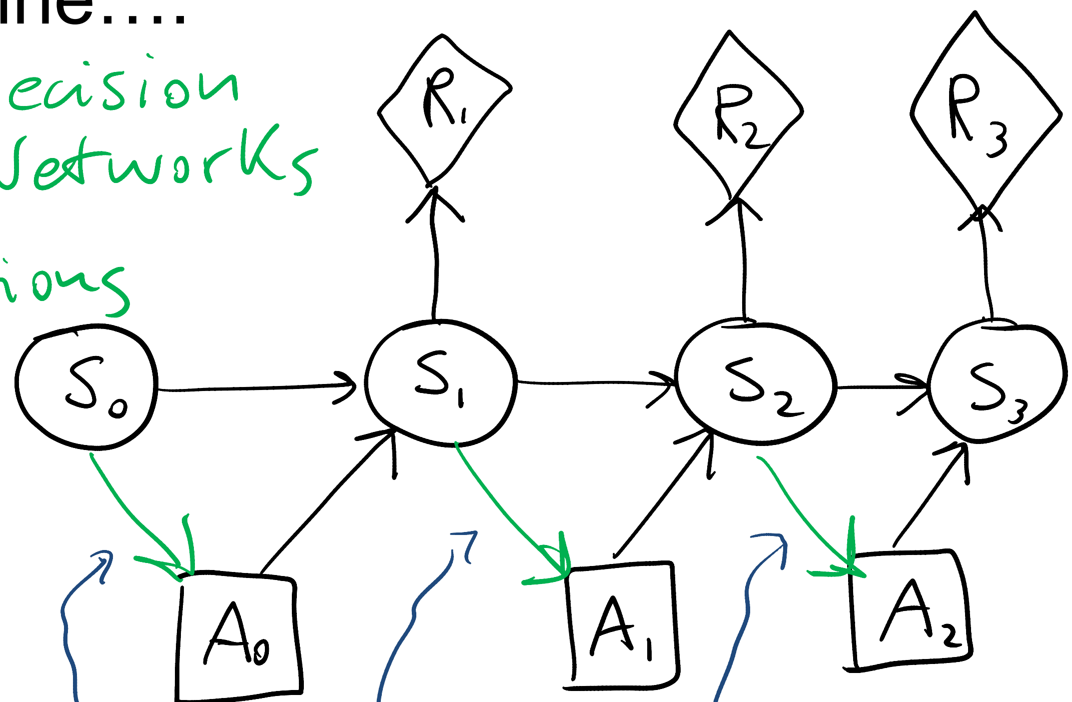
To manage an ongoing (indefinite… infinite) decision process, we combine….

Markov Chains & Decision Networks

Markovian

Stationary

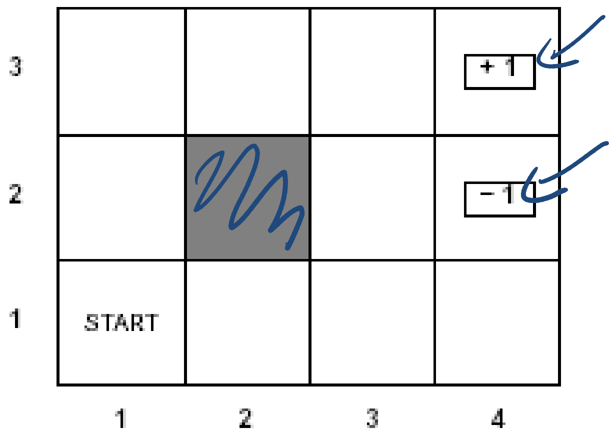
Assumptions

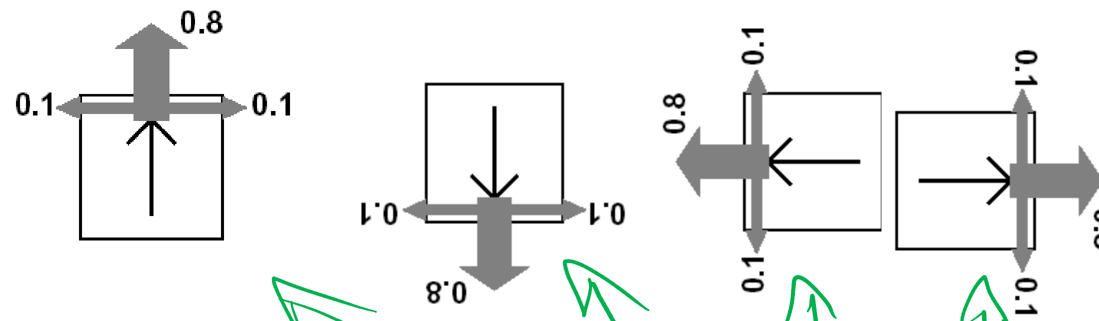


Utility not just at the end
BUT
Sequence of rewards

Fully Observable

Example MDP: Scenario and Actions





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

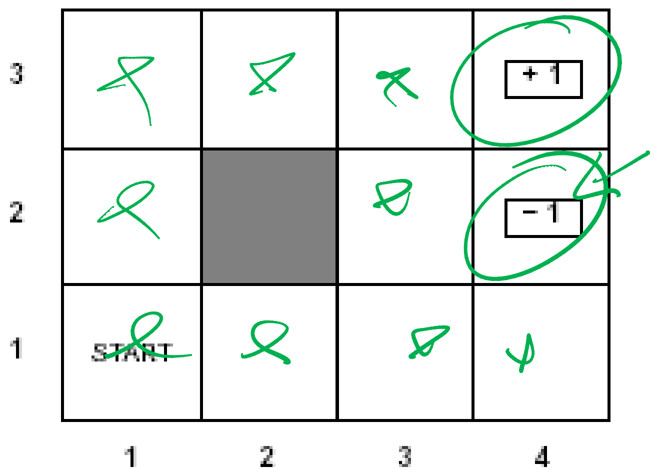
Each action has:

- 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 {(11), (12), ... , (24), (34)}

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

Example MDP: Rewards

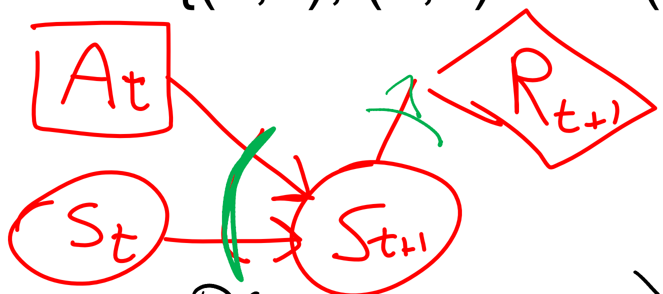


$$R(s) = \begin{cases} -0.04 & \text{(small penalty) for nonterminal states} \\ \pm 1 & \text{for terminal states} \end{cases}$$

Example MDP: Underlying info structures

Four actions *Up, Down, Left, Right*

Eleven States: {(1,1), (1,2)…… (3,4)}



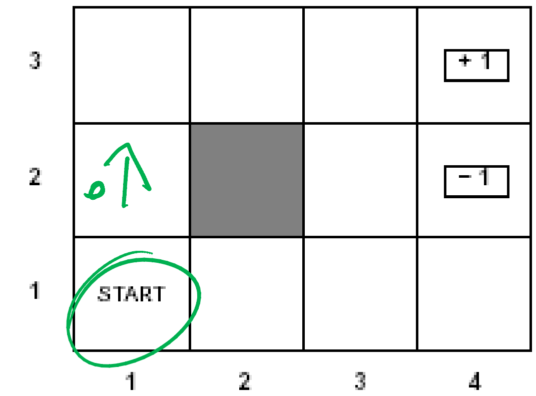


Table $4 \times 11 \times 11$ $P(S_{t+1} \mid S_t, A_t)$

Up

	1,1	2,1	1,2	3,1 …
1,1	.1	.8	.1	0000
2,1	0	.2	0	.8
⋮				

Down

L

R

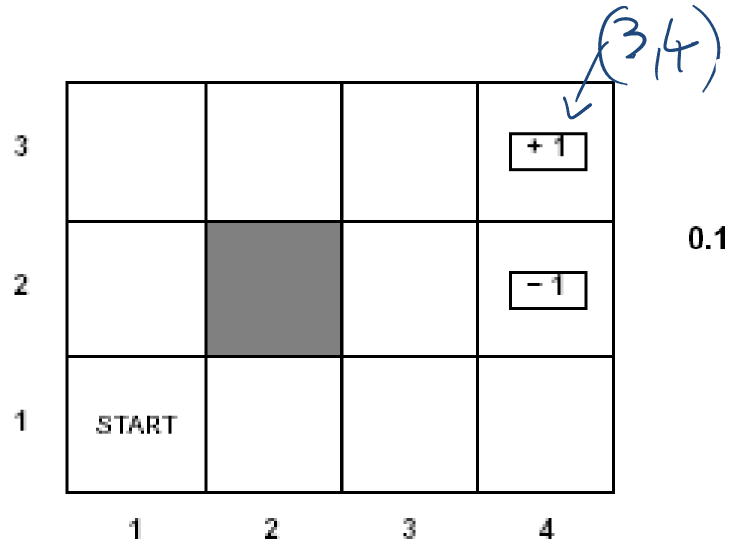
$P(S_0)$

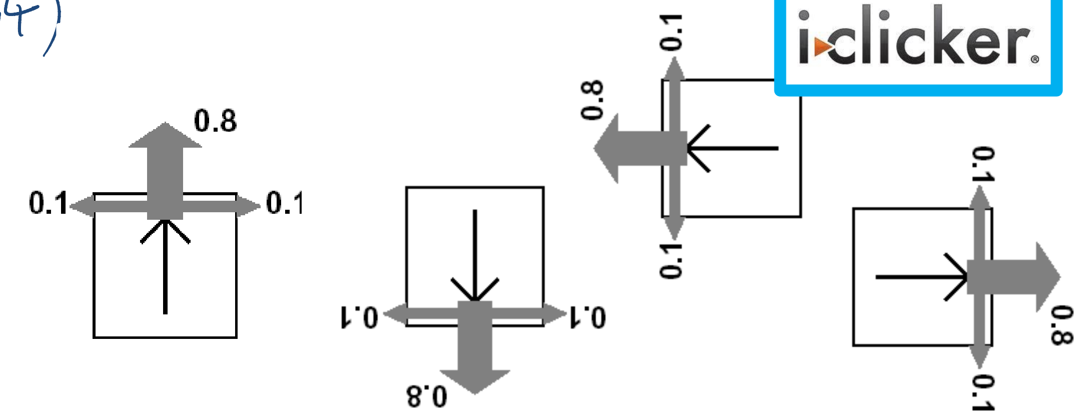
1,1	1
⋮	0
	0
	0
	0
	0
	0

$R(s)$

1,1	−.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*

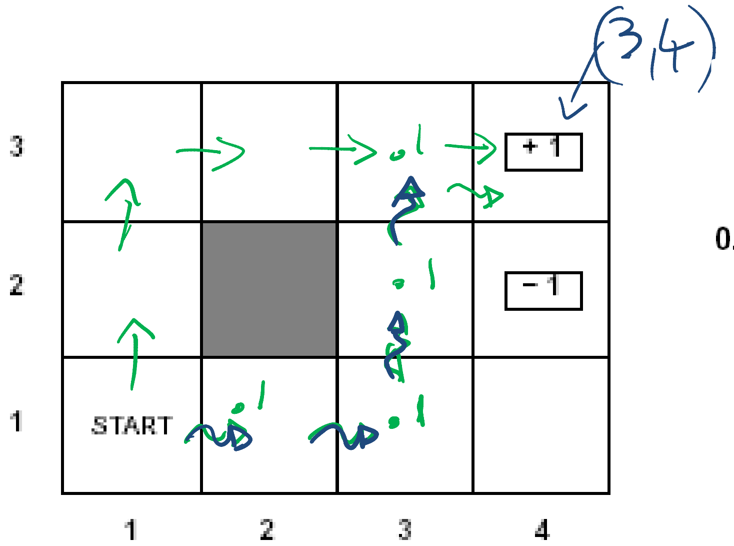
With what probability?

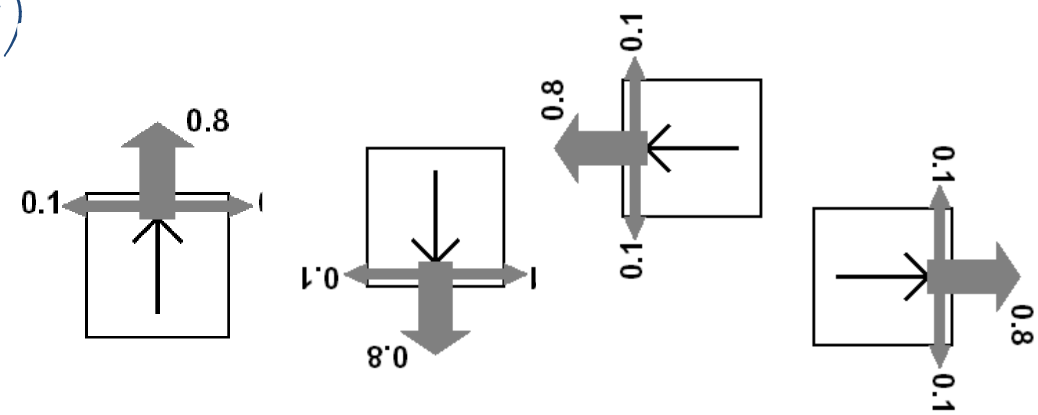
A. $(0.8)^5$

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

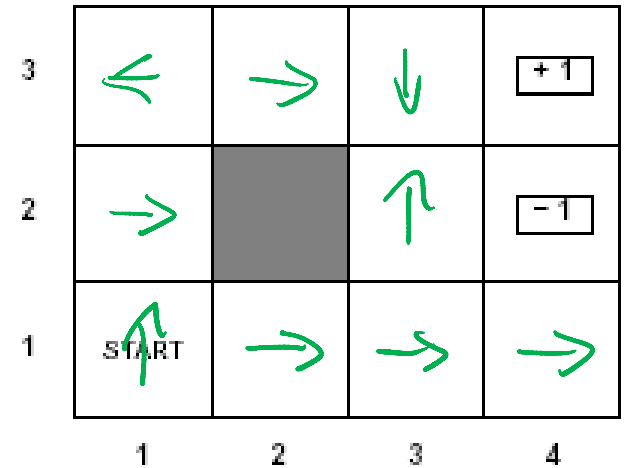
MDPs: Policy

- 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?

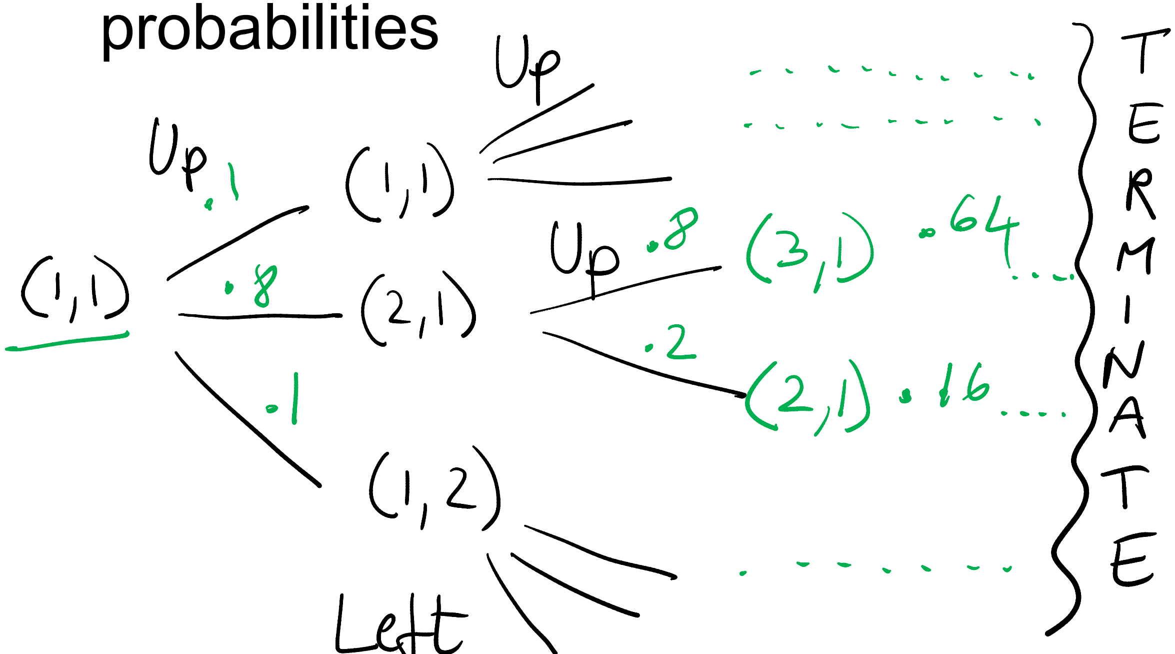
policy

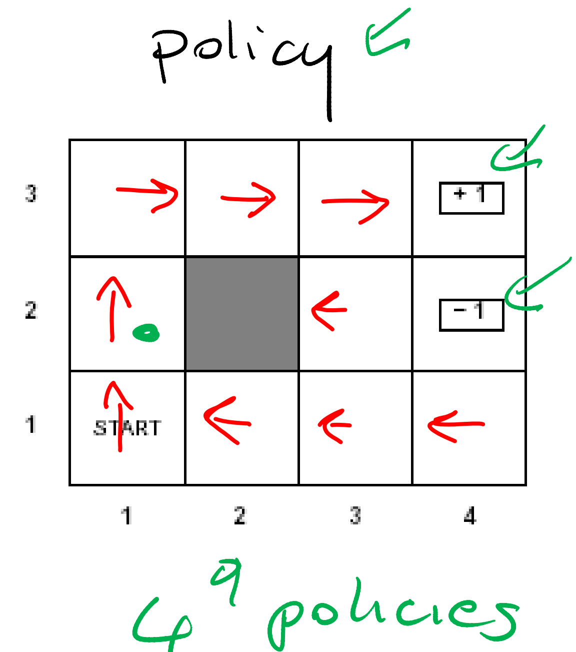
  - So a policy for an MDP is a single decision function $\pi(s)$ that specifies what the agent should do for each state s



How to evaluate a policy

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

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

$+.72$

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

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

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

- $V^{\pi}(s)$: 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.

Can we express $Q^{\pi}(s, a)$ in terms of $V^{\pi}(s)$?

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

A.

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

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

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

$$R_{\max} \text{ bound on R(s) for every } s$$

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

$$= \sum_{i=0}^{\infty} \gamma^i r_{i+1} \leq \sum_{i=0}^{\infty} \gamma^i R_{\max} = \frac{R_{\max}}{1-\gamma}$$

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

$$Q^{\pi}(s, a) = \underbrace{R(s)}_{\text{reward obtained in } s} + \underbrace{\gamma}_{\text{Discount factor}} \underbrace{\sum_{s'}}_{\text{states reachable from } s \text{ by doing } a} \underbrace{P(s' \mid s, a)}_{\text{Probability of getting to } s' \text{ from } s \text{ via } a} \underbrace{V^{\pi}(s')}_{\text{expected value of following policy } \pi \text{ in } s'}$$

Value of a policy and Optimal policy

We can also compute $V^{\pi}(s)$ in terms of $Q^{\pi}(s, a)$

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

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

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

STOP HERE

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

$V^{\pi^*}(s)$

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

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

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

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

- 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