Decision Processes

CISC 7404 - Decision Making

Steven Morad

University of Macau

Review 2
Markov Processes 5
Exercise 16
Markov Control Processes 18
Markov Decision Processes 25
Exercise 40
Coding 42
Exam Next Class 47

Review

Review

Last time, we reviewed probability and bandits

Review

Last time, we reviewed probability and bandits

Review

Markov Processes

Markov Processes

Decisions must make some change in the world

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Some things we can model using Markov processes:

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Some things we can model using Markov processes:

- Music

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Some things we can model using Markov processes:

- Music
- DNA sequences

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Some things we can model using Markov processes:

- Music
- DNA sequences
- Cryptography

Markov Processes

Decisions must make some change in the world

If they make no change, they do not matter, and are not decisions!

Before we look at decision making, let us think about how we model change in the world

Markov processes are a popular way to model the world

Some things we can model using Markov processes:
- Music
- DNA sequences
- Cryptography
- History

Markov Processes

We can model almost anything as a Markov process

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

A Markov process consists of two parts

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

A Markov process consists of two parts

The **state space**

$$S$$

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

A Markov process consists of two parts

The **state space**	The **state transition function**
S	$T : S \mapsto \Delta S$

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

A Markov process consists of two parts

The **state space**	The **state transition function**
S	$T : S \mapsto \Delta S$
	$T(s) = \Pr(s' \mid s)$

Markov Processes

We can model almost anything as a Markov process

So what is a Markov process?

It is a probabilistic model of dynamical systems, where the **future depends on the past**

A Markov process consists of two parts

The **state space**

$$S$$

The **state transition function**

$$T : S \mapsto \Delta S$$

$$T(s) = \Pr(s' \mid s)$$

Let us do an example to understand this

Markov Processes

Problem: Predict the weather

Markov Processes

Problem: Predict the weather

$$S = \{\text{rain}, \text{cloud}, \text{sun}\} = \{R, C, S\}$$

Markov Processes

Problem: Predict the weather

$$S = \{\text{rain}, \text{cloud}, \text{sun}\} = \{R, C, S\}$$

$$\begin{bmatrix} \Pr(C \mid C) & \Pr(R \mid C) & \Pr(S \mid C) \\ \Pr(C \mid R) & \Pr(R \mid R) & \Pr(S \mid R) \\ \Pr(C \mid S) & \Pr(R \mid S) & \Pr(S \mid S) \end{bmatrix}$$

$$= \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.5 & 0.3 & 0.2 \\ 0.5 & 0.1 & 0.4 \end{bmatrix}$$

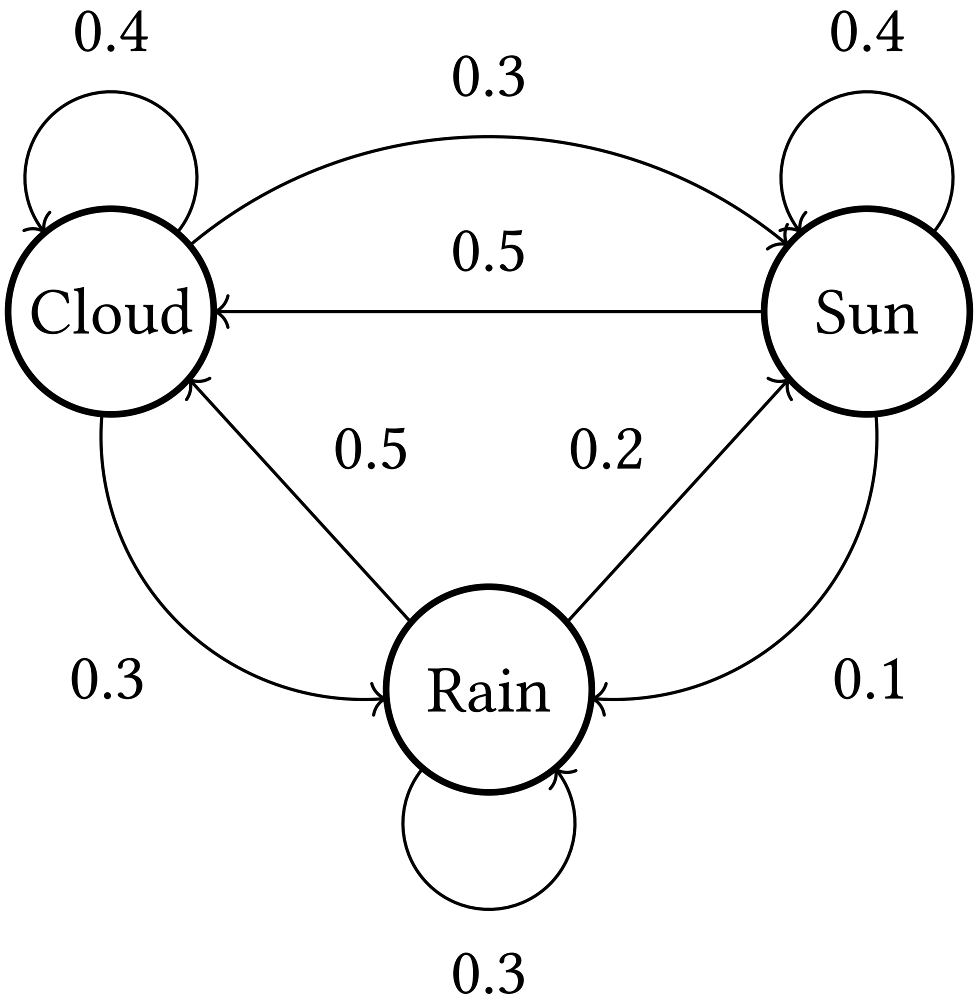
Markov Processes

Problem: Predict the weather

$$S = \{\text{rain}, \text{cloud}, \text{sun}\} = \{R, C, S\}$$

$$\begin{bmatrix} \Pr(C \mid C) & \Pr(R \mid C) & \Pr(S \mid C) \\ \Pr(C \mid R) & \Pr(R \mid R) & \Pr(S \mid R) \\ \Pr(C \mid S) & \Pr(R \mid S) & \Pr(S \mid S) \end{bmatrix}$$

$$= \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.5 & 0.3 & 0.2 \\ 0.5 & 0.1 & 0.4 \end{bmatrix}$$

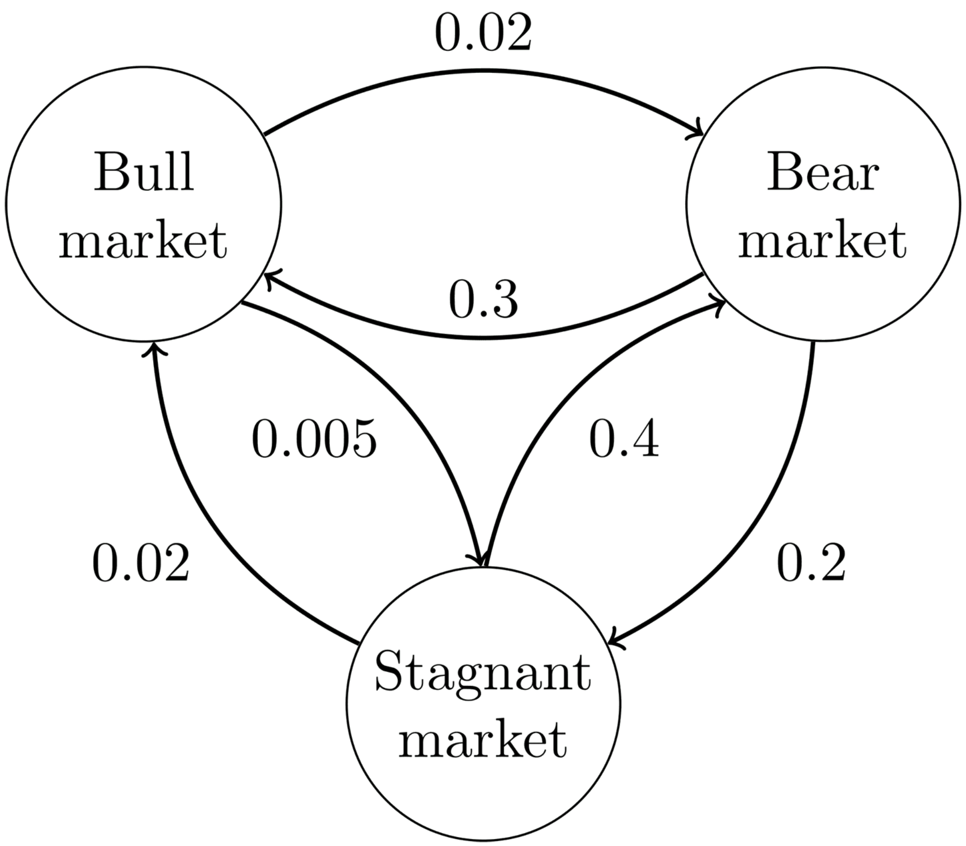


Markov Processes

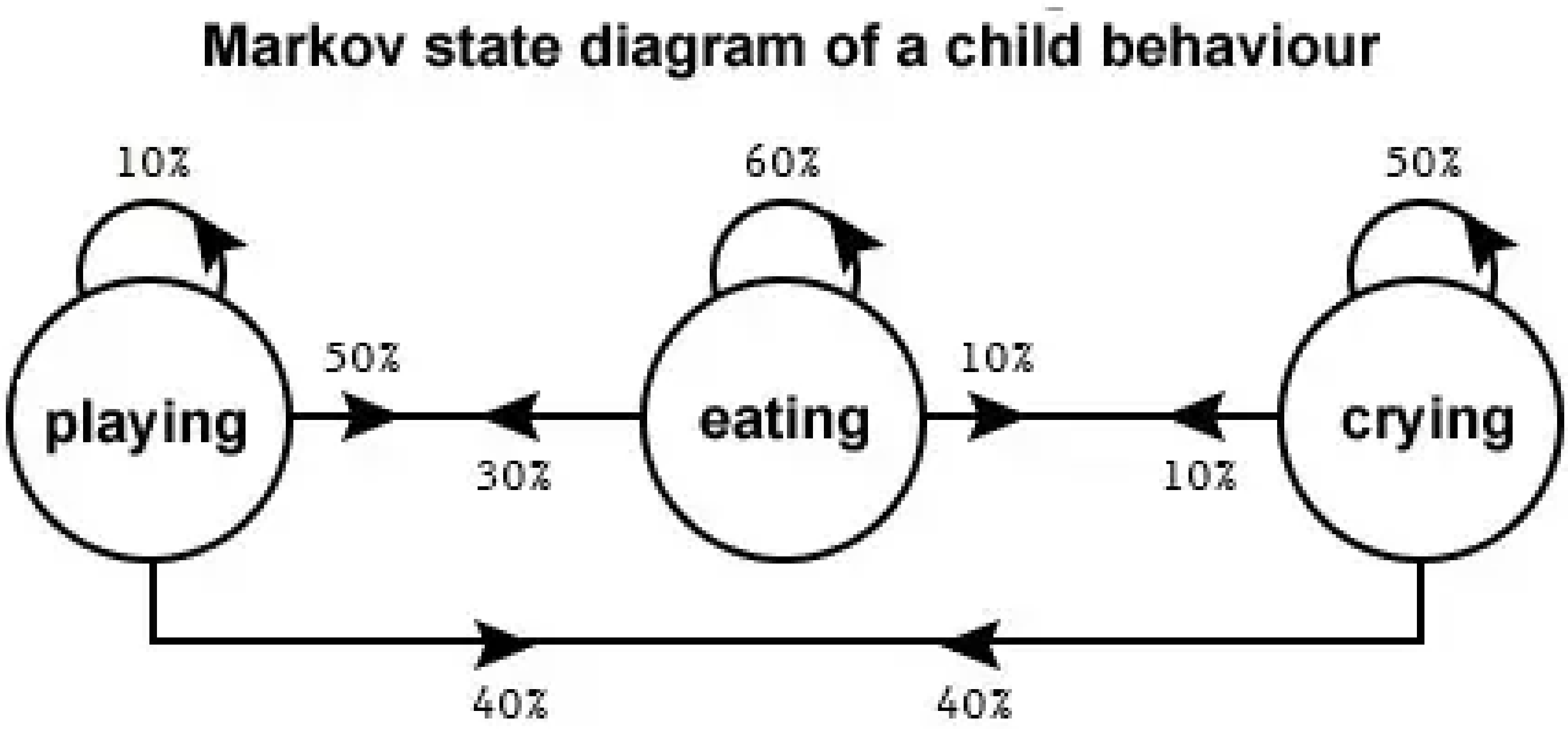
Of course, we can model many other systems as Markov processes

Markov Processes

Of course, we can model many other systems as Markov processes



Markov Processes



Markov Processes

Why is it called a **Markov** process?

Markov Processes

Why is it called a **Markov** process?

It follows the **Markov** property:

The next state only depends on the current state

Markov Processes

Why is it called a **Markov** process?

It follows the **Markov** property:

The next state only depends on the current state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

Markov Processes

Why is it called a **Markov** process?

It follows the **Markov** property:

The next state only depends on the current state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

This is a very important condition we must always satisfy

Markov Processes

Why is it called a **Markov** process?

It follows the **Markov** property:

The next state only depends on the current state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

This is a very important condition we must always satisfy

If we cannot satisfy it, then the process is **not** Markov

Markov Processes

Why is it called a **Markov** process?

It follows the **Markov** property:

The next state only depends on the current state

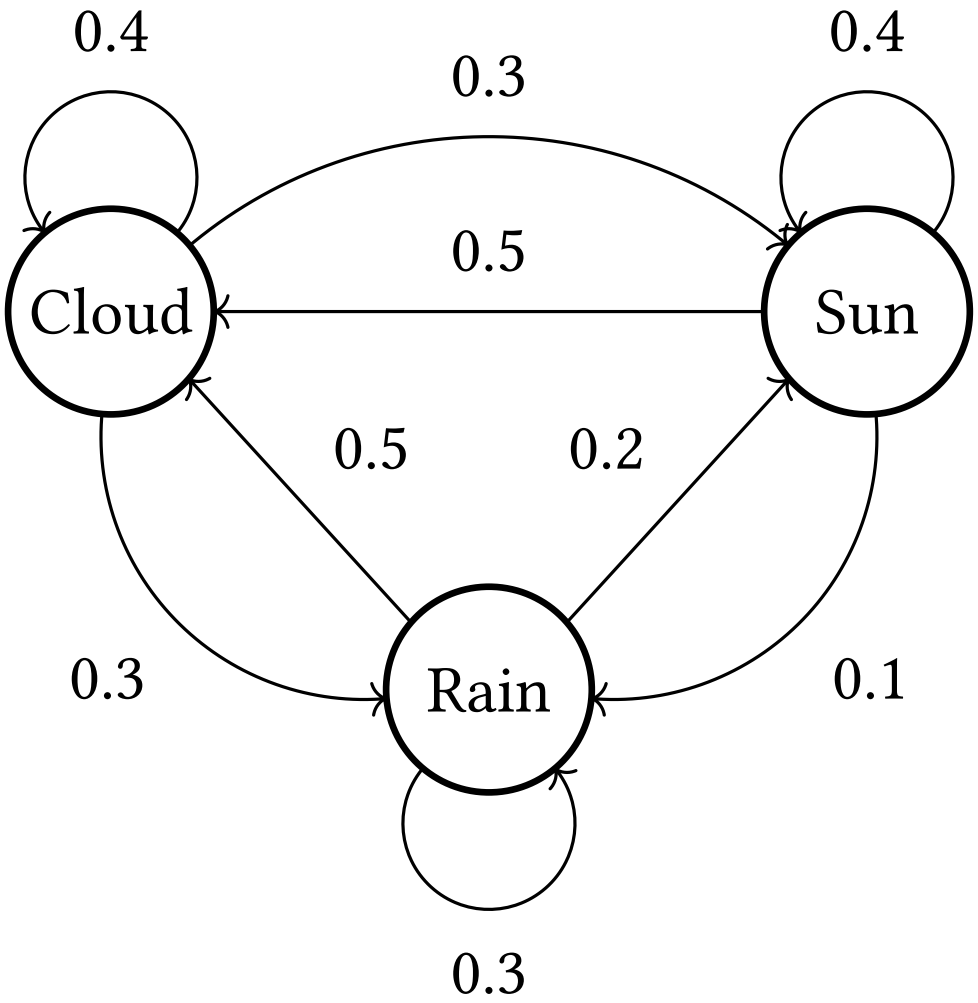
$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

This is a very important condition we must always satisfy

If we cannot satisfy it, then the process is **not** Markov

Markov Processes

To compute the next node, we only look at the current node



Markov Processes

Question: When does a Markov process end?

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end

- Dying in a video game

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end

- Dying in a video game
- Winning a video game

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end

- Dying in a video game
- Winning a video game
- Running out of money

Markov Processes

Question: When does a Markov process end?

Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end

- Dying in a video game
- Winning a video game
- Running out of money

Question: How can we model this?

Markov Processes

Question: When does a Markov process end?

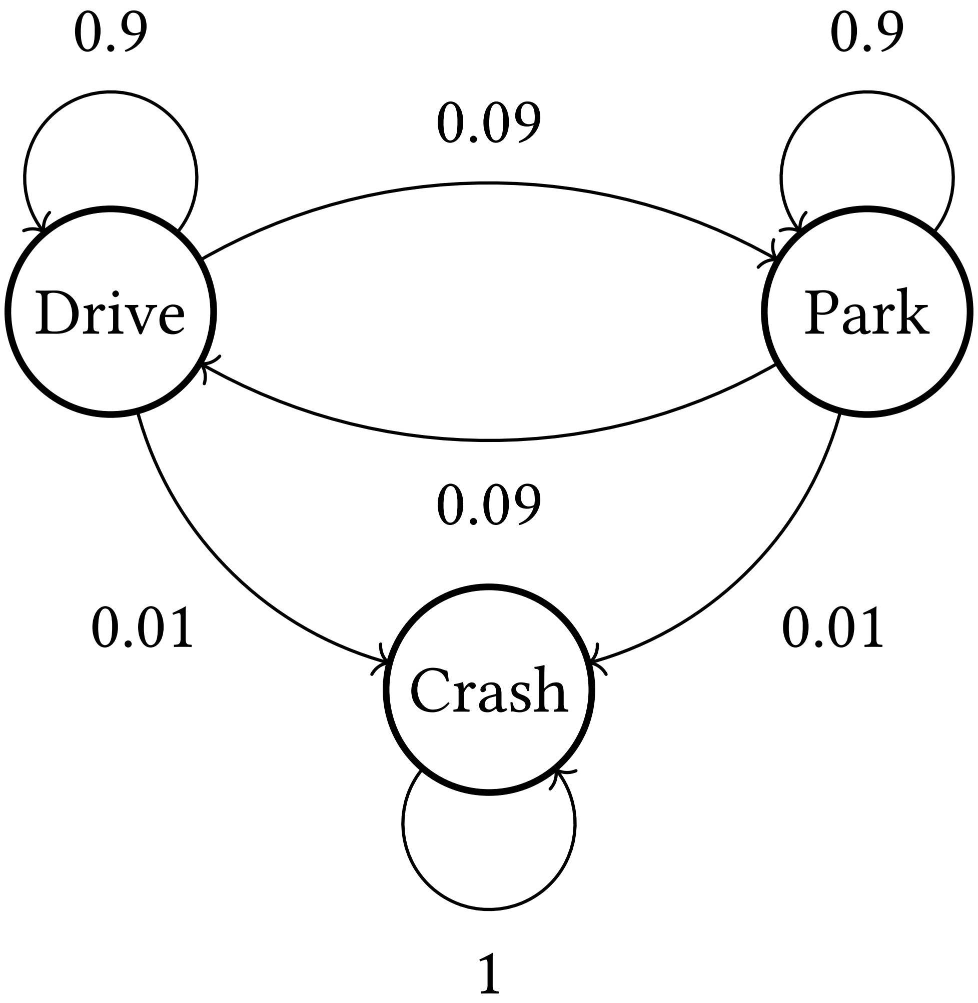
Answer: Technically, they never end. You are always in a specific state

However, many processes we like to model eventually end
- Dying in a video game
- Winning a video game
- Running out of money

Question: How can we model this?

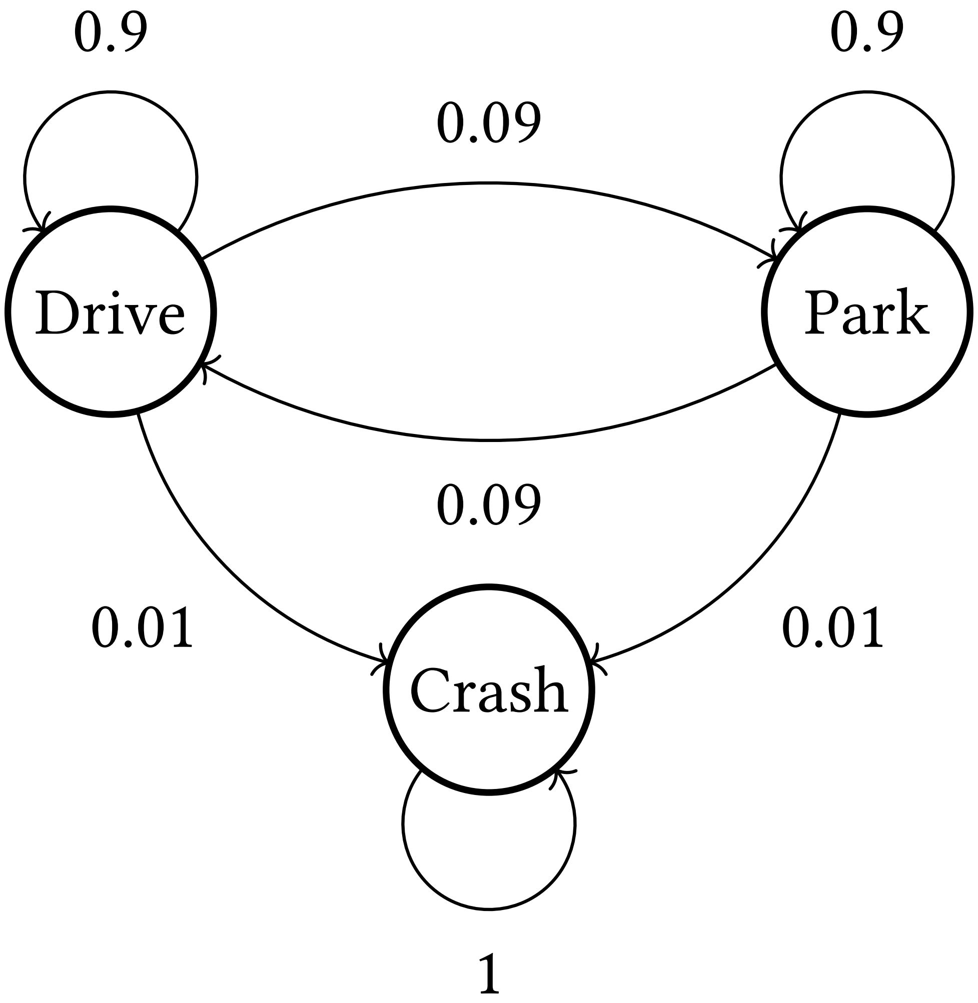
Answer: We create a **terminal state** that we cannot leave

Markov Processes



Upon reaching a terminal state, we get stuck

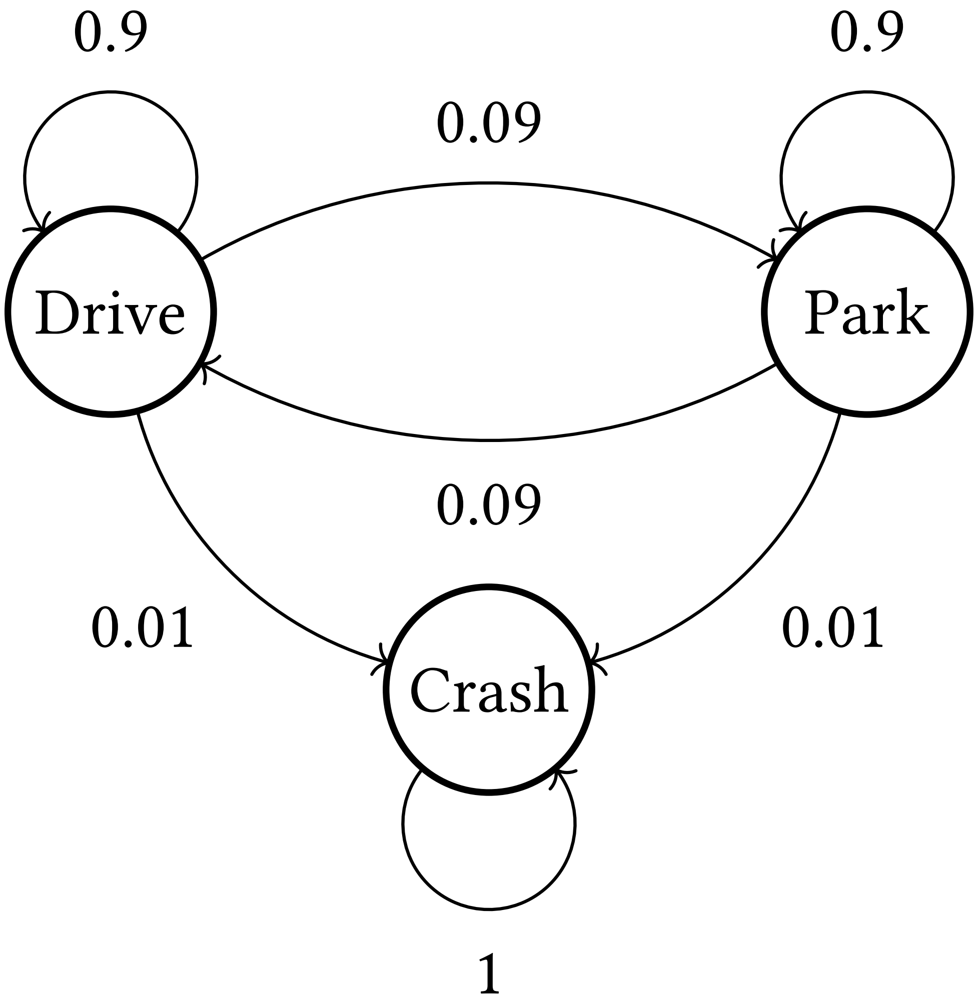
Markov Processes



Upon reaching a terminal state, we get stuck

Once we crash our car, we cannot drive or park any more

Markov Processes



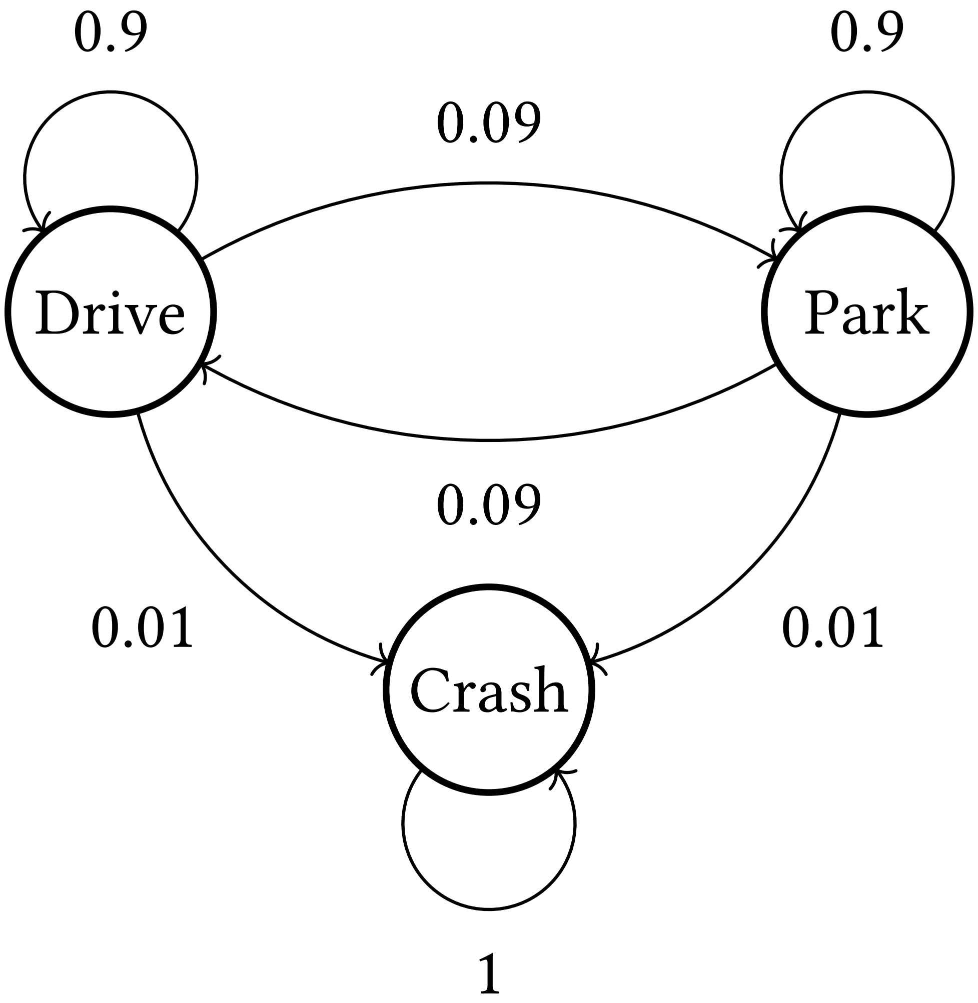
Upon reaching a terminal state, we get stuck

Once we crash our car, we cannot drive or park any more

The only transition from a terminal state is back to itself

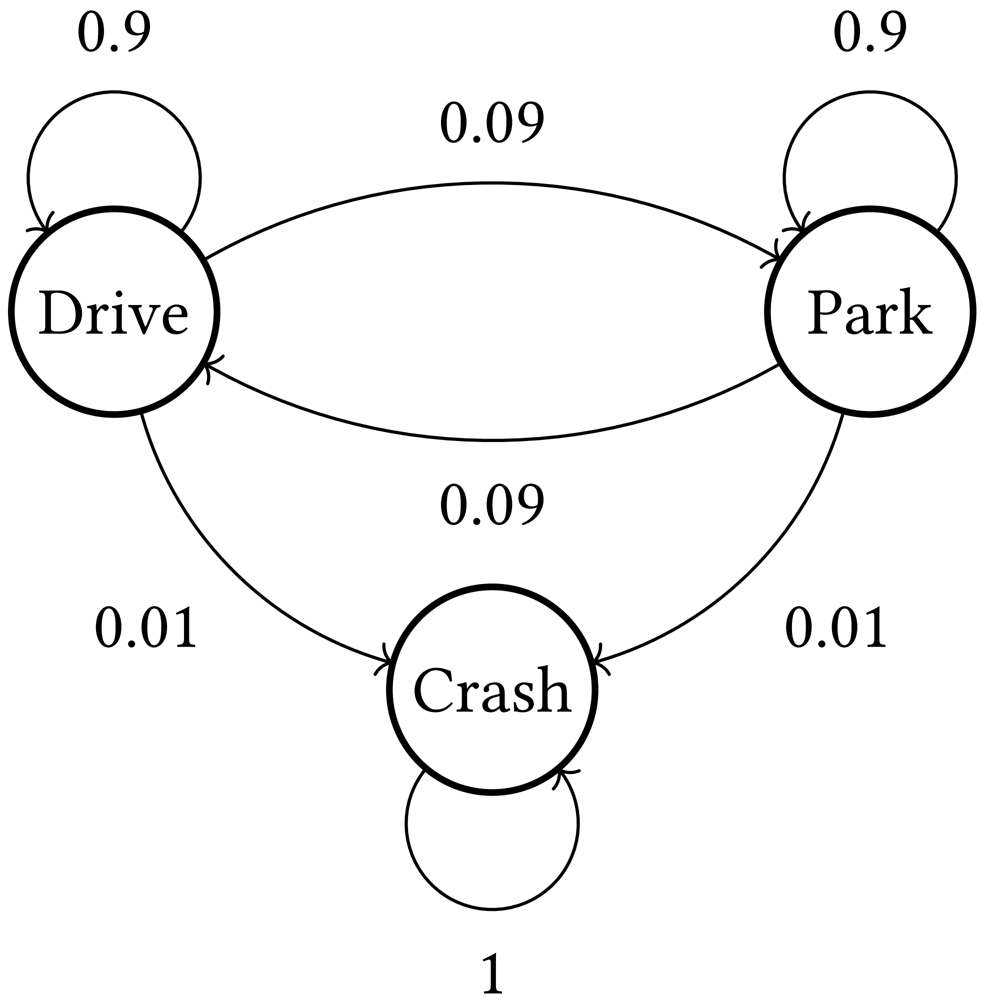
$$\Pr(s' = s_{\text{terminal}} \mid s = s_{\text{terminal}}) = 1.0$$

Markov Processes



We call the sequence of states until the terminal state an **episode**

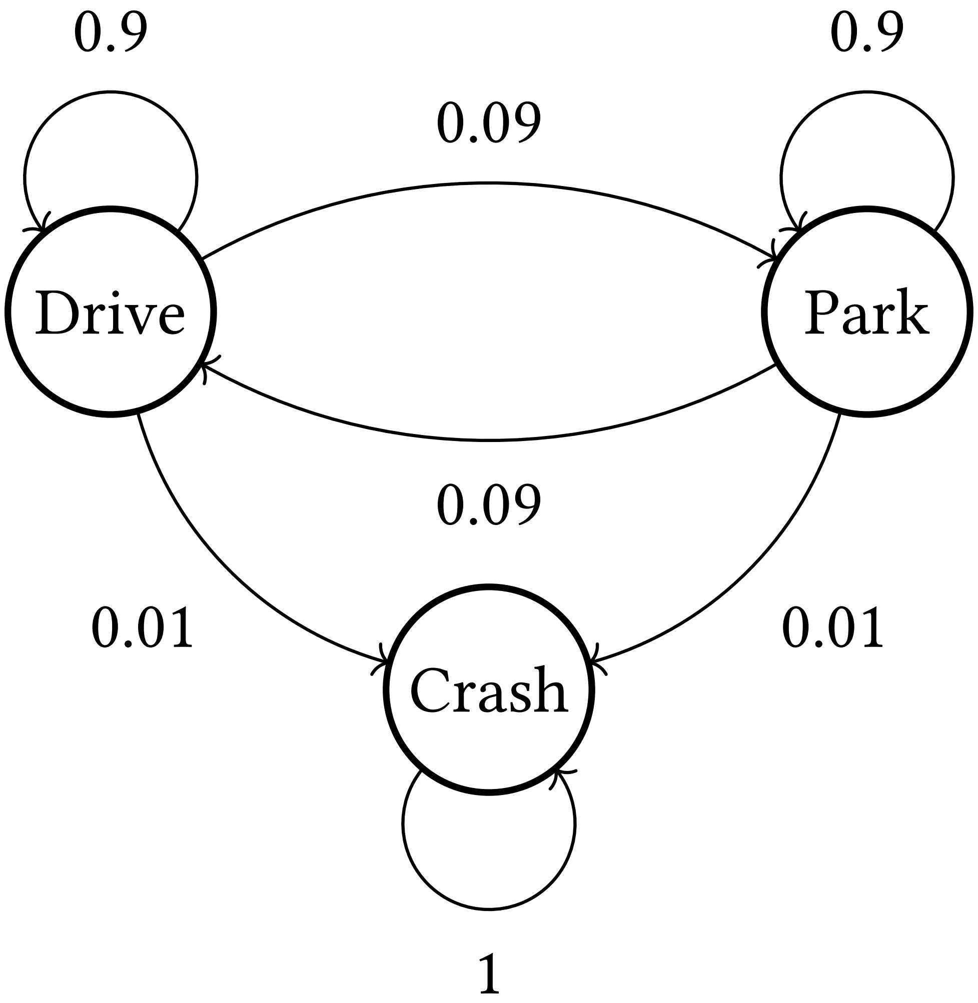
Markov Processes



We call the sequence of states until the terminal state an **episode**

$$\begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix}$$

Markov Processes



We call the sequence of states until the terminal state an **episode**

$$\begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix} = \begin{bmatrix} \text{Drive} \\ \text{Drive} \\ \text{Park} \\ \vdots \\ \text{Crash} \end{bmatrix}$$

Exercise

Exercise

Design an MDP about a problem you care about

Exercise

Design an MDP about a problem you care about

- 3 or more states

Exercise

Design an MDP about a problem you care about

- 3 or more states
- State transition function $T = \Pr(s' \mid s)$ for all s, s'

Exercise

Design an MDP about a problem you care about

- 3 or more states
- State transition function $T = \Pr(s' \mid s)$ for all s, s'
- Create a terminal state

Markov Control Processes

Markov Control Processes

Question: How can we model decision making in a Markov process?

Markov Control Processes

Question: How can we model decision making in a Markov process?

Answer: We can't (yet)

Markov Control Processes

Question: How can we model decision making in a Markov process?

Answer: We can't (yet)

Markov processes follow the state transition function T, there are no decisions for us to make

Markov Control Processes

Question: How can we model decision making in a Markov process?

Answer: We can't (yet)

Markov processes follow the state transition function T, there are no decisions for us to make

We will modify the Markov process for decision making

Markov Control Processes

A Markov process models the predetermined evolution of some system

Markov Control Processes

A Markov process models the predetermined evolution of some system

We call this system the **environment**, because we cannot control it

Markov Control Processes

A Markov process models the predetermined evolution of some system

We call this system the **environment**, because we cannot control it

For decisions to matter, they must change the environment

Markov Control Processes

A Markov process models the predetermined evolution of some system

We call this system the **environment**, because we cannot control it

For decisions to matter, they must change the environment

We introduce the **agent** to make decisions that change the environment

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

Markov process

$$(S, T)$$

$$T : S \mapsto \Delta S$$

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

Markov process	Markov control process
(S, T)	(S, A, T)
$T : S \mapsto \Delta S$	$T : S \times A \mapsto \Delta S$

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

Markov process	Markov control process
(S, T)	(S, A, T)
$T : S \mapsto \Delta S$	$T : S \times A \mapsto \Delta S$

In a Markov process, the future follows a specified evolution

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

Markov process	Markov control process
(S, T)	(S, A, T)
$T : S \mapsto \Delta S$	$T : S \times A \mapsto \Delta S$

In a Markov process, the future follows a specified evolution

In a Markov control process, we can control the evolution!

Markov Control Processes

The agent takes **actions** $a \in A$ that change the environment

The action space A defines what our agent can do

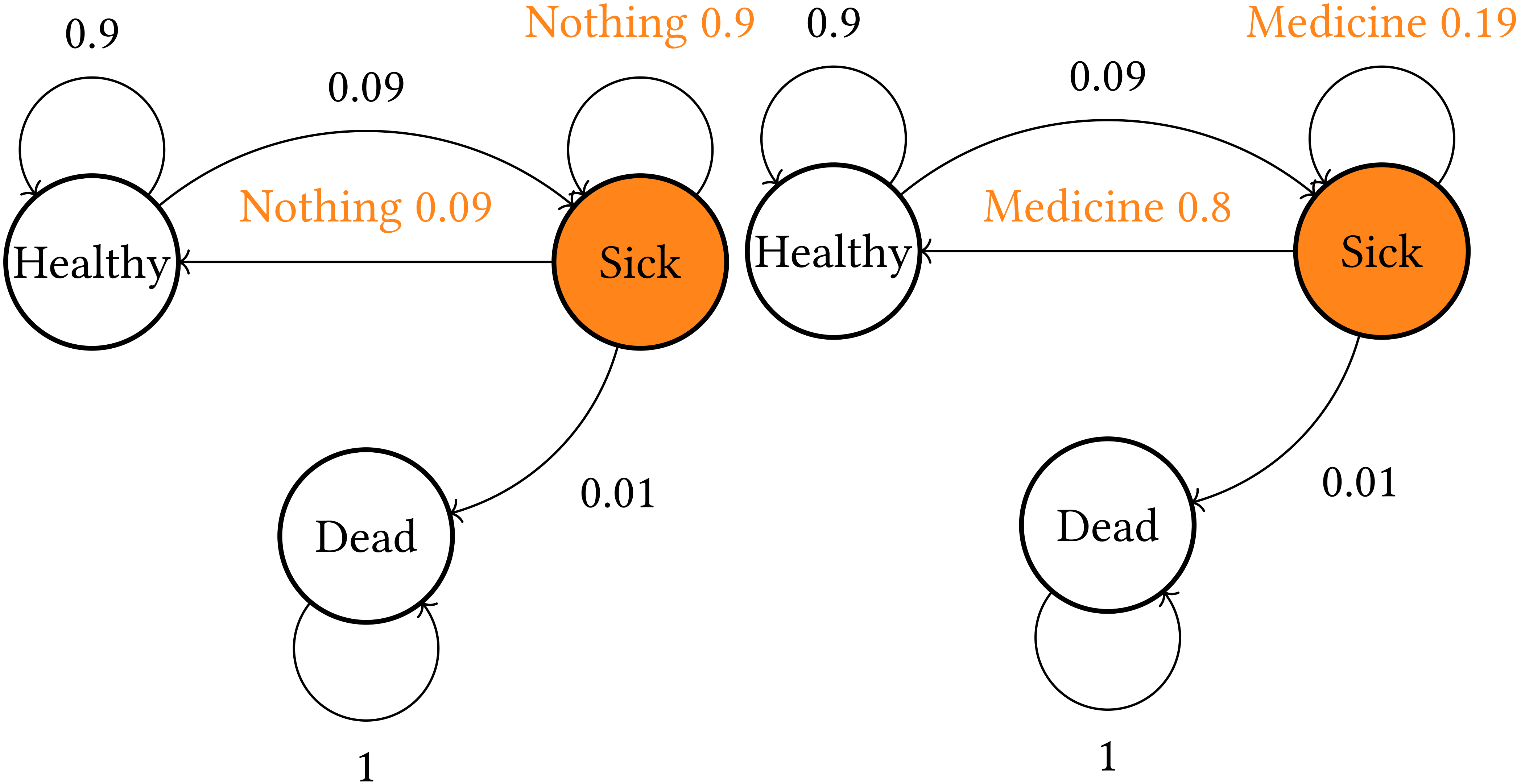
Markov process	Markov control process
(S, T)	(S, A, T)
$T : S \mapsto \Delta S$	$T : S \times A \mapsto \Delta S$

In a Markov process, the future follows a specified evolution

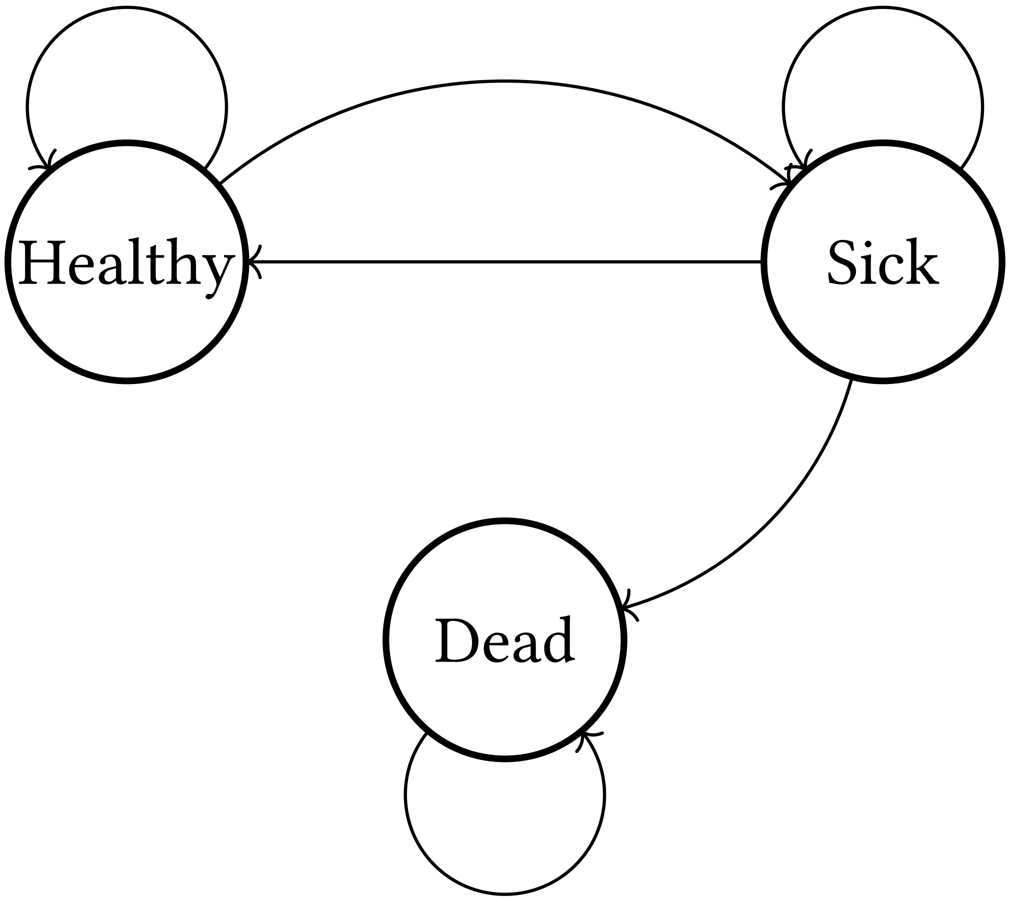
In a Markov control process, we can control the evolution!

Let us see an example

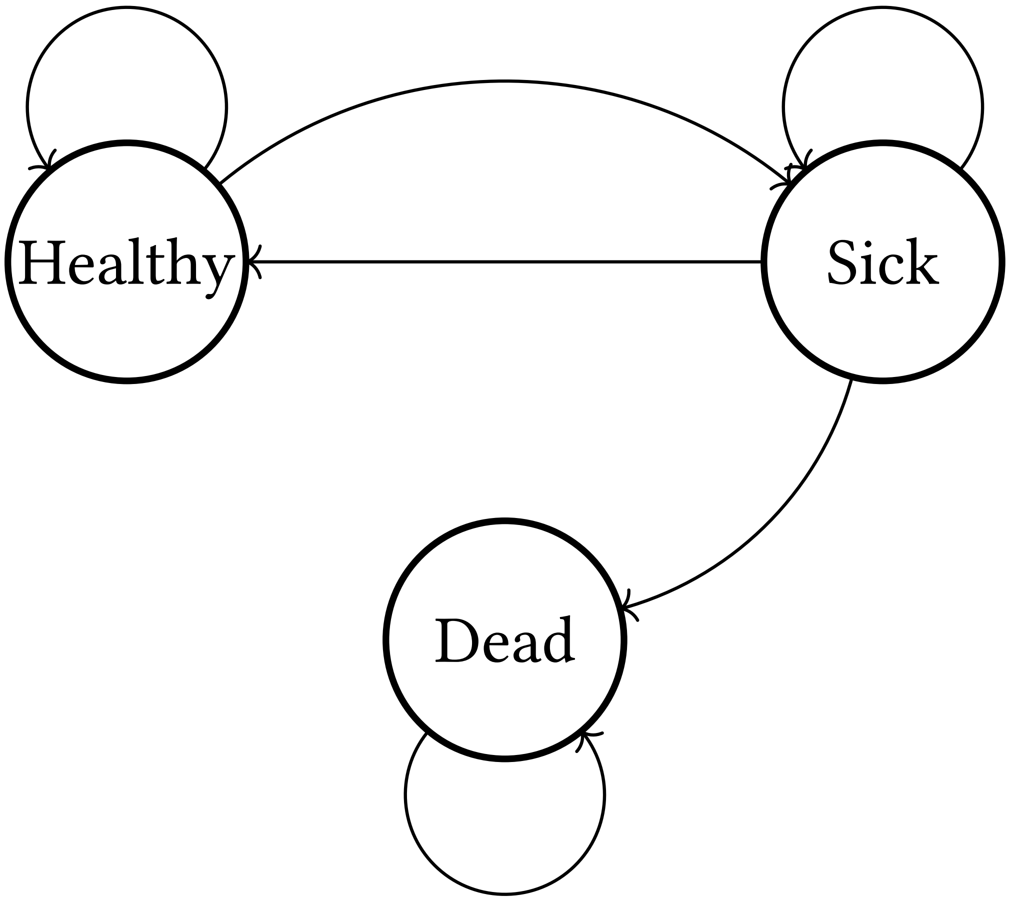
Markov Control Processes



Markov Control Processes

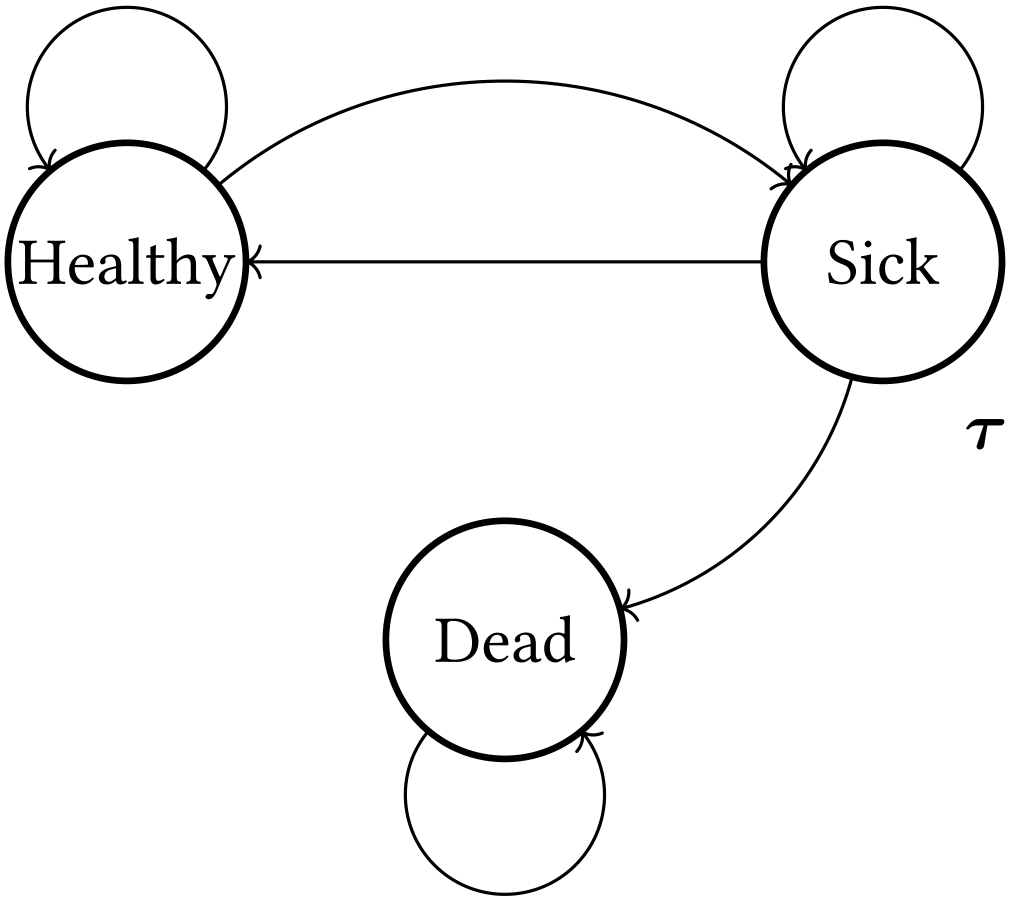


Markov Control Processes



The **trajectory** contains the states and actions until a terminal state

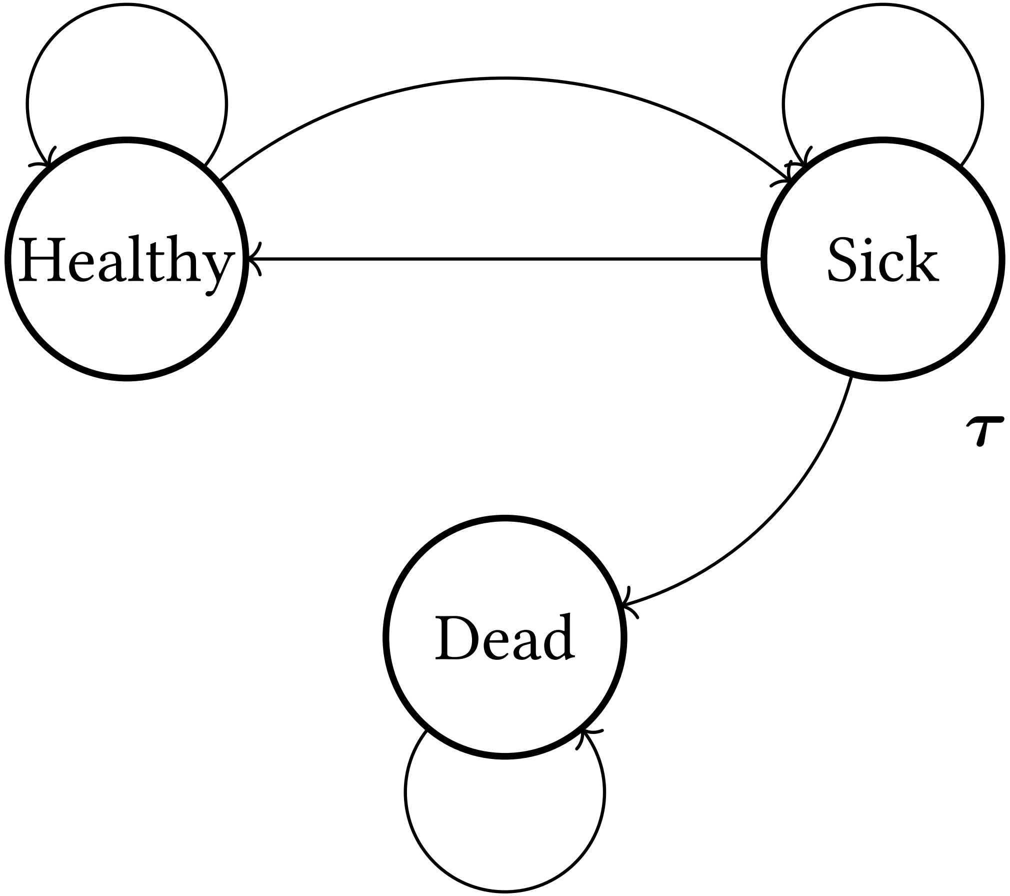
Markov Control Processes



The **trajectory** contains the states and actions until a terminal state

$$\boldsymbol{\tau} = \begin{bmatrix} s_0 & a_0 \\ s_1 & a_1 \\ s_2 & a_2 \\ \vdots & \vdots \\ s_n & \emptyset \end{bmatrix}$$

Markov Control Processes



The **trajectory** contains the states and actions until a terminal state

$$\boldsymbol{\tau} = \begin{bmatrix} s_0 & a_0 \\ s_1 & a_1 \\ s_2 & a_2 \\ \vdots & \vdots \\ s_n & \emptyset \end{bmatrix} = \begin{bmatrix} \text{Healthy} & \text{Nothing} \\ \text{Sick} & \text{Nothing} \\ \text{Sick} & \text{Medicine} \\ \vdots & \vdots \\ \text{Dead} & \emptyset \end{bmatrix}$$

Markov Control Processes

Markov control processes let us control which states we visit

Markov Control Processes

Markov control processes let us control which states we visit

They do not tell us which states are good to visit

Markov Control Processes

Markov control processes let us control which states we visit

They do not tell us which states are good to visit

How can we make optimal decisions if we cannot tell how good a decision is?

Markov Control Processes

Markov control processes let us control which states we visit

They do not tell us which states are good to visit

How can we make optimal decisions if we cannot tell how good a decision is?

We need something to tell us how good it is to be in a state!

Markov Decision Processes

Markov Decision Processes

Markov decision processes (MDPs) add a measure of “goodness” to Markov control processes

Markov Decision Processes

Markov decision processes (MDPs) add a measure of "goodness" to Markov control processes

We use a **reward function** R to measure the goodness of being in a specific state

Markov Decision Processes

Markov decision processes (MDPs) add a measure of “goodness” to Markov control processes

We use a **reward function** R to measure the goodness of being in a specific state

Sutton and Barto:

$$R : S \times A \mapsto \mathbb{R}$$

Markov Decision Processes

Markov decision processes (MDPs) add a measure of "goodness" to Markov control processes

We use a **reward function** R to measure the goodness of being in a specific state

Sutton and Barto:	This course:
$R : S \times A \mapsto \mathbb{R}$	$R : S \mapsto \mathbb{R}$

Markov Decision Processes

Markov process

$$(S, T)$$

$$T : S \mapsto \Delta S$$

Markov Decision Processes

Markov process

$$(S, T)$$

$$T : S \mapsto \Delta S$$

Markov control process

$$(S, A, T)$$

$$T : S \times A \mapsto \Delta S$$

Markov Decision Processes

Markov process

$$(S, T)$$

$$T : S \mapsto \Delta S$$

Markov control process

$$(S, A, T)$$

$$T : S \times A \mapsto \Delta S$$

Markov decision process

$$(S, A, T, R, \gamma)$$

$$T : S \times A \mapsto \Delta S$$

$$R : S \mapsto \mathbb{R}$$

Markov Decision Processes

The **history** contains the states, actions, and rewards until termination

Markov Decision Processes

The **history** contains the states, actions, and rewards until termination

$$\boldsymbol{H} = \begin{bmatrix} s_0 & a_0 & r_0 \\ s_1 & a_1 & r_1 \\ \vdots & \vdots & \vdots \\ s_n & \emptyset & r_n \end{bmatrix}$$

Markov Decision Processes

We want to maximize the reward

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

$$s_d = \text{Dumpling} \qquad s_n = \text{Noodle}$$

$$R(s_d) = 10 \qquad R(s_n) = 15$$

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

$$s_d = \text{Dumpling} \qquad s_n = \text{Noodle}$$

$$R(s_d) = 10 \qquad R(s_n) = 15$$

Result: Eat noodle

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

$$s_d = \text{Dumpling} \qquad s_n = \text{Noodle}$$

$$R(s_d) = 10 \qquad R(s_n) = 15$$

Result: Eat noodle

$$R(s_d) = 5 \qquad R(s_n) = -3$$

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

$$s_d = \text{Dumpling} \qquad s_n = \text{Noodle}$$

$$R(s_d) = 10 \qquad R(s_n) = 15$$

Result: Eat noodle

$$R(s_d) = 5 \qquad R(s_n) = -3$$

Result: Eat dumpling

Markov Decision Processes

We want to maximize the reward

The reward function determines the agent behavior

$$s_d = \text{Dumpling} \qquad\qquad s_n = \text{Noodle}$$

$R(s_d) = 10$ $\quad$ $R(s_n) = 15$ $\quad$ **Result:** Eat noodle

$R(s_d) = 5$ $\quad$ $R(s_n) = -3$ $\quad$ **Result:** Eat dumpling

We can write this mathematically as

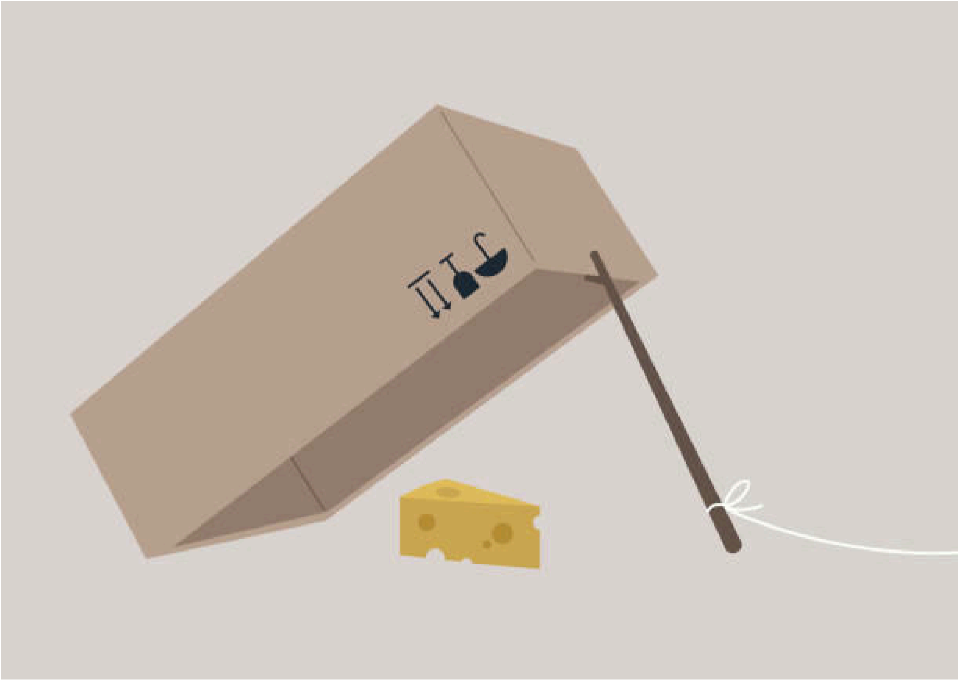
$$\arg\max_{s \in S} R(s)$$

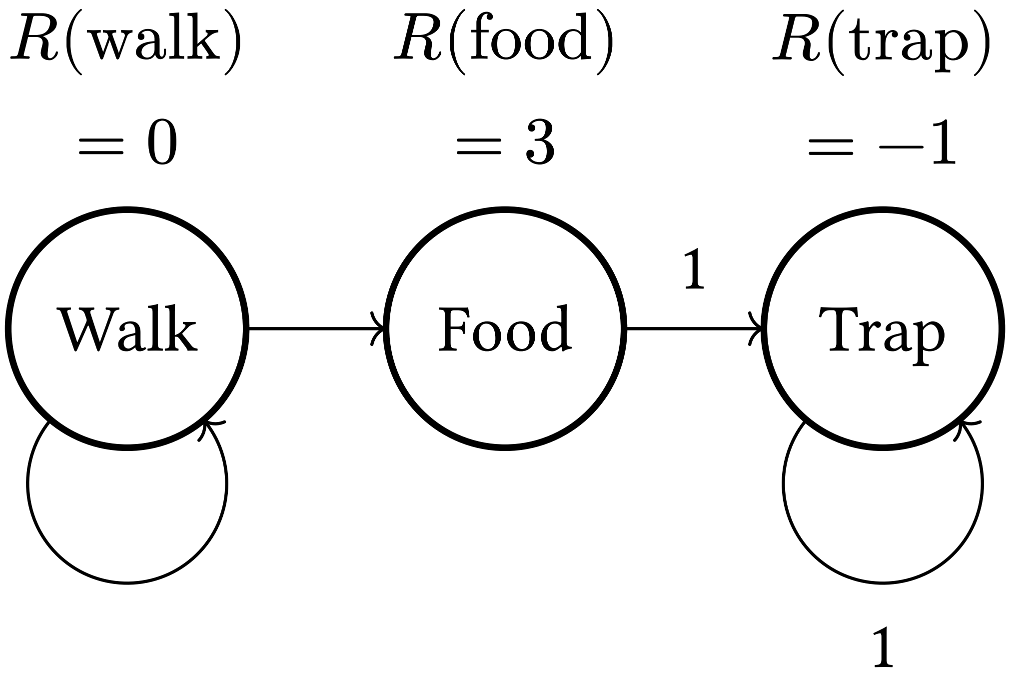
Markov Decision Processes

However, maximizing the reward is not always ideal

Markov Decision Processes

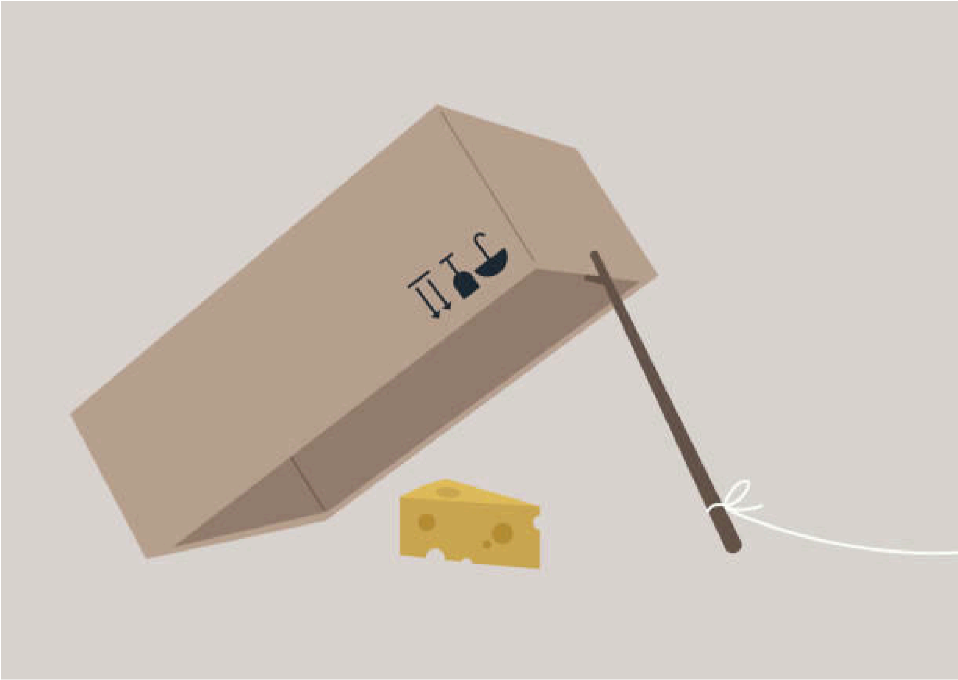
However, maximizing the reward is not always ideal

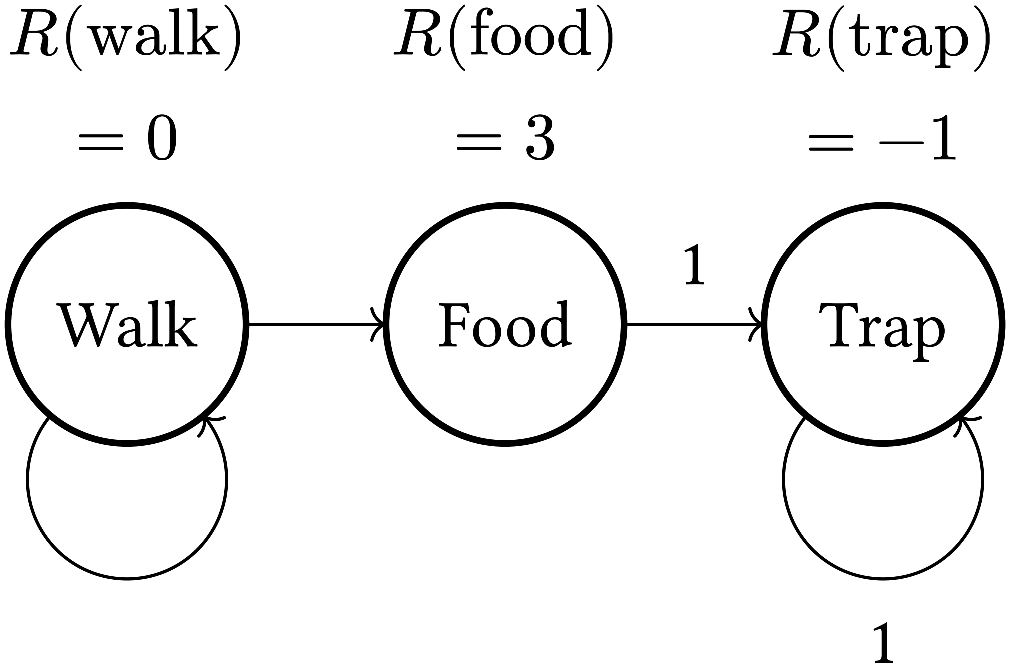




Markov Decision Processes

However, maximizing the reward is not always ideal

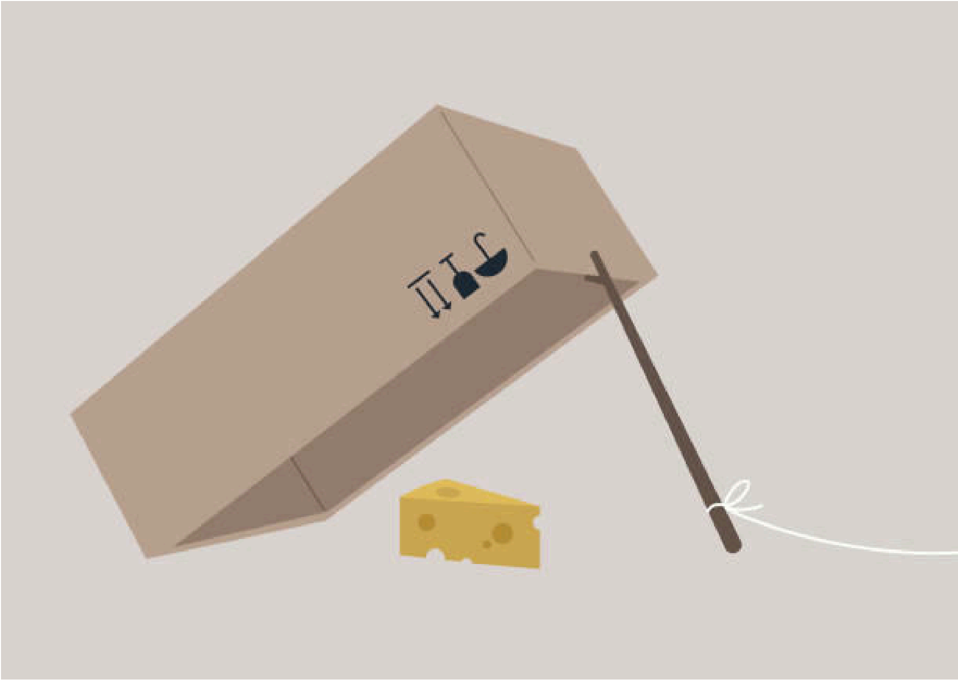


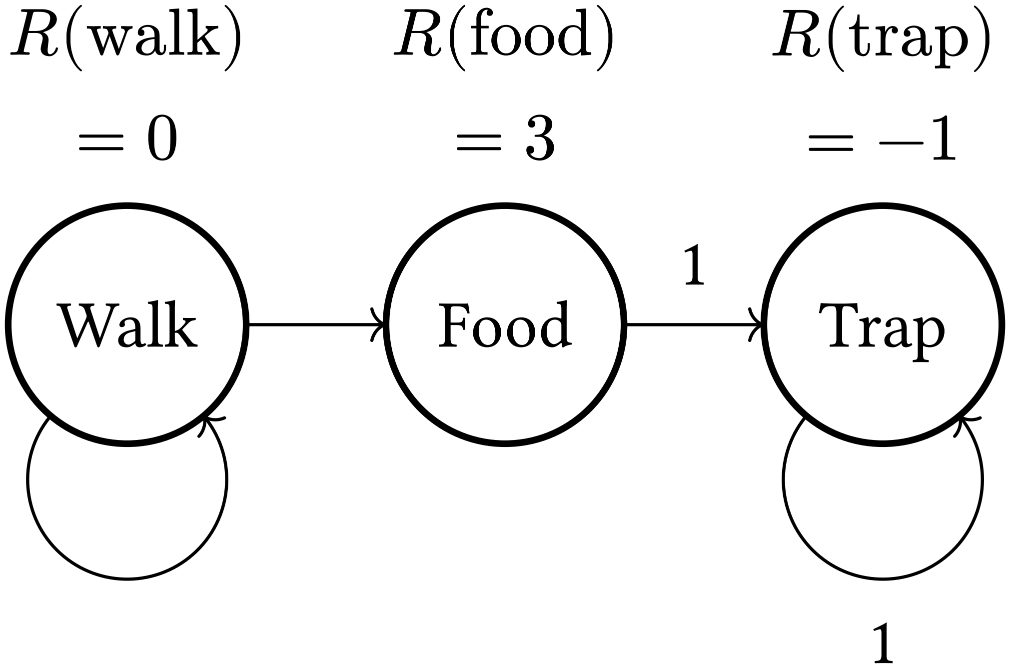


$$\arg\max_{s \in S} R(s)$$

Markov Decision Processes

However, maximizing the reward is not always ideal

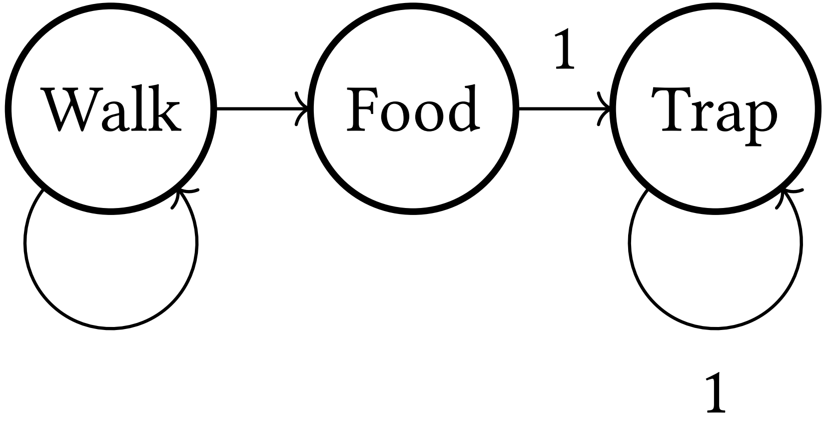




$$\arg\max_{s \in S} R(s) = \text{food}$$

Markov Decision Processes

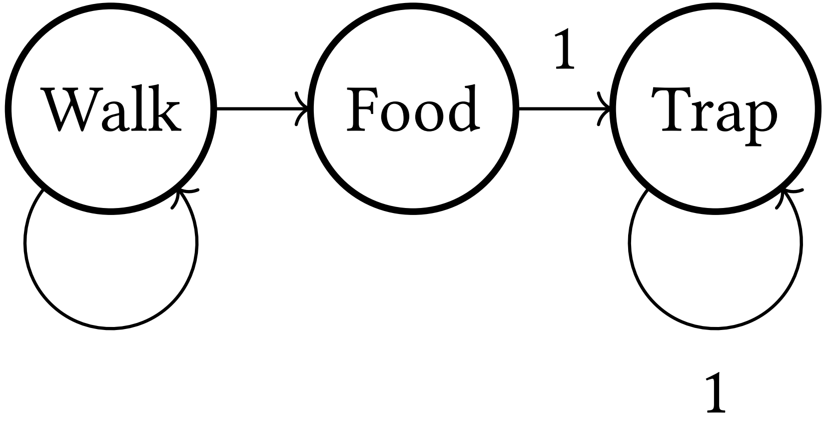
$R(\text{walk}) = 0 \quad R(\text{food}) = 3 \quad R(\text{trap}) = -1$



Instead, we maximize the **sum** of rewards

Markov Decision Processes

$R(\text{walk}) = 0$ $\quad R(\text{food}) = 3$ $\quad R(\text{trap}) = -1$

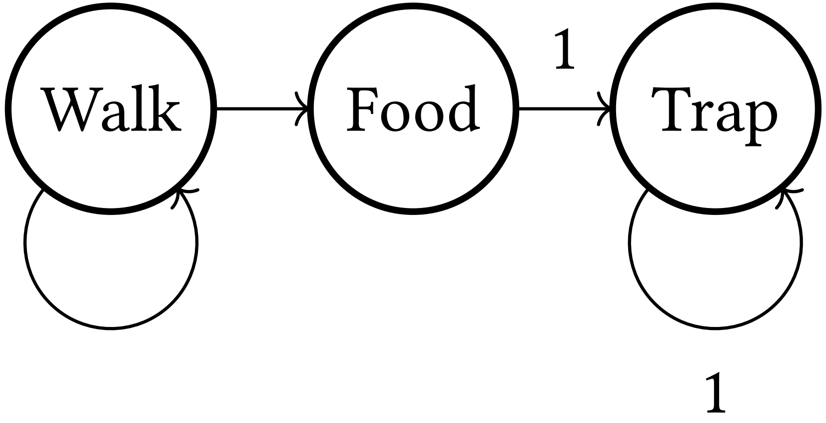


Instead, we maximize the **sum** of rewards

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Markov Decision Processes

$R(\text{walk}) = 0$ $R(\text{food}) = 3$ $R(\text{trap}) = -1$



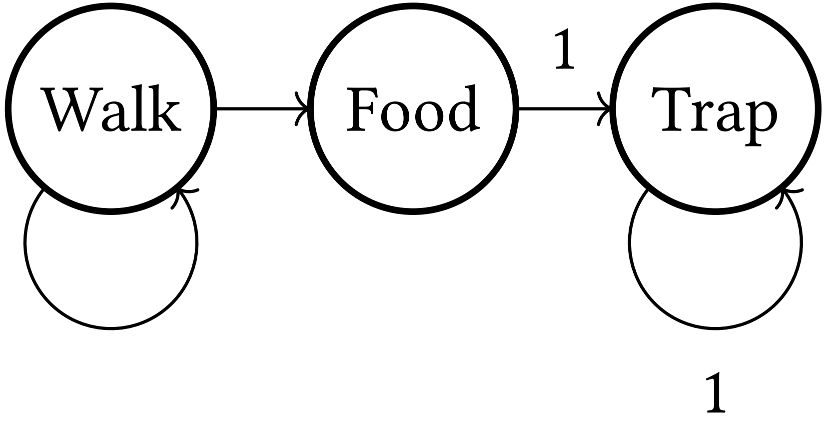
Instead, we maximize the **sum** of rewards

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We call this the **return**

Markov Decision Processes

$R(\text{walk}) = 0$ $\quad R(\text{food}) = 3$ $\quad R(\text{trap}) = -1$



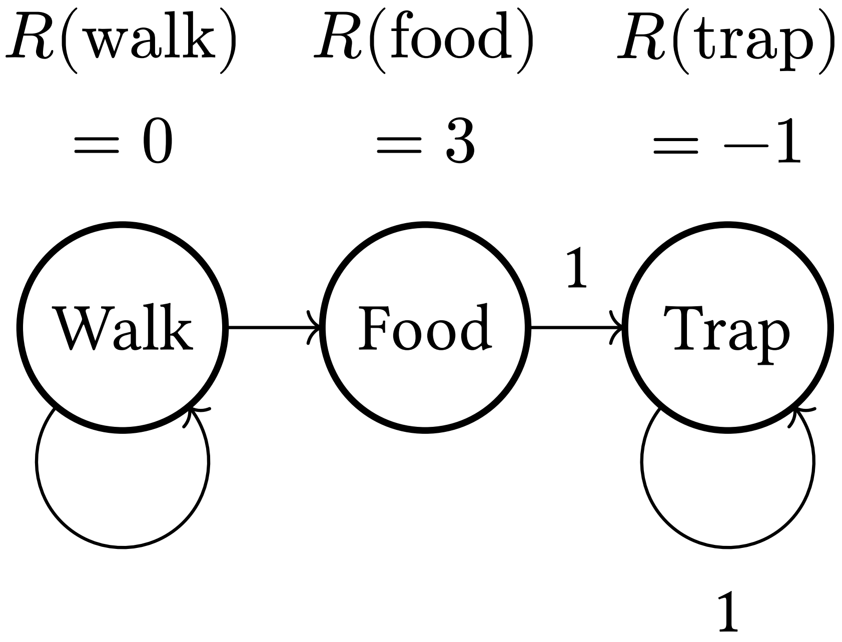
Instead, we maximize the **sum** of rewards

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We call this the **return**

$$R(\text{walk}) + R(\text{walk}) + R(\text{walk}) + \ldots = 0 + 0 + \ldots \qquad = 0$$

Markov Decision Processes



Instead, we maximize the **sum** of rewards

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

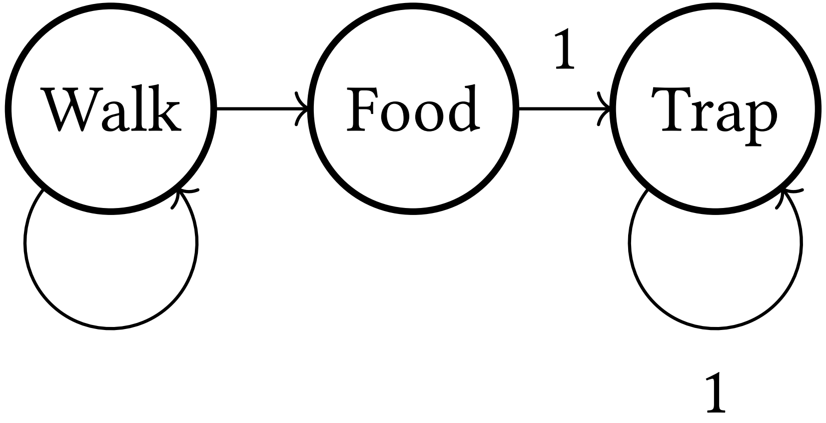
We call this the **return**

$$R(\text{walk}) + R(\text{walk}) + R(\text{walk}) + \ldots = 0 + 0 + \ldots = 0$$

$$R(\text{food}) + R(\text{trap}) + R(\text{trap}) + \ldots = 3 - 1 - 1 - \ldots = -\infty$$

Markov Decision Processes

$R(\text{walk}) = 0$ $R(\text{food}) = 3$ $R(\text{trap}) = -1$



Instead, we maximize the **sum** of rewards

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We call this the **return**

$$R(\text{walk}) + R(\text{walk}) + R(\text{walk}) + \dots = 0 + 0 + \dots = 0$$

$$R(\text{food}) + R(\text{trap}) + R(\text{trap}) + \dots = 3 - 1 - 1 - \dots = -\infty$$

Now, we make better decisions!

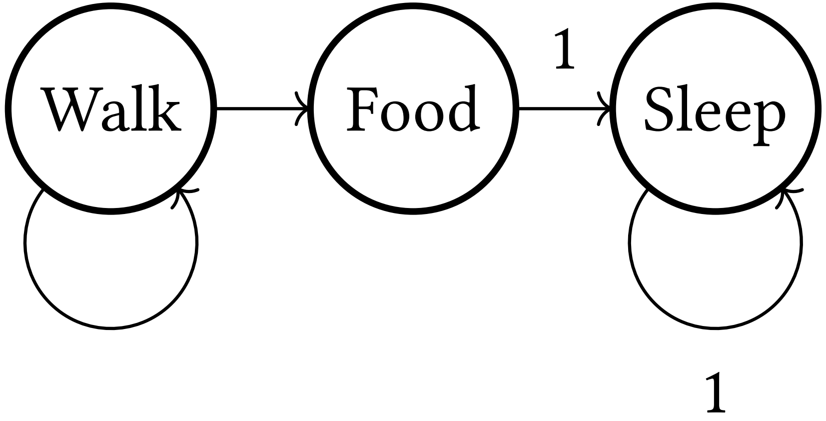
Markov Decision Processes

Consider one more example

Markov Decision Processes

Consider one more example

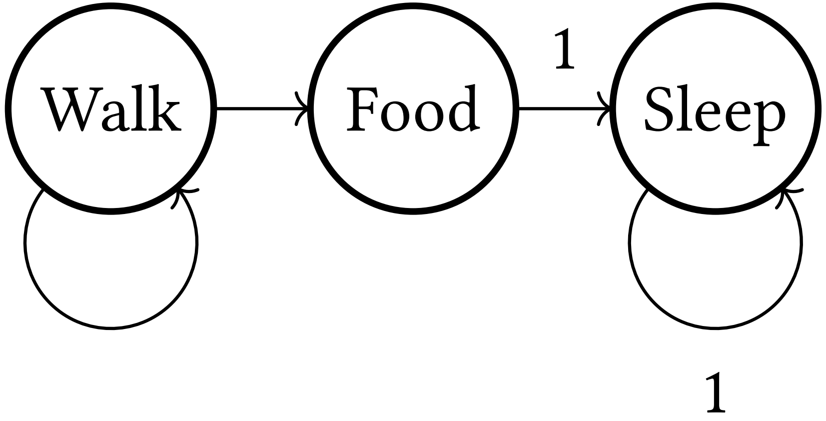
$R(\text{walk}) = 0 \quad R(\text{food}) = 3 \quad R(\text{sleep}) = 0$



Markov Decision Processes

Consider one more example

$R(\text{walk}) = 0 \quad R(\text{food}) = 3 \quad R(\text{sleep}) = 0$

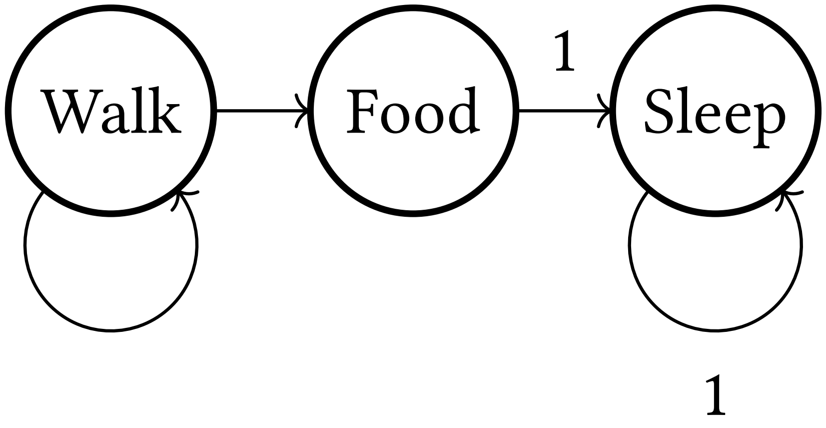


Question: What is the optimal sequence of states?

Markov Decision Processes

Consider one more example

$R(\text{walk}) = 0$ $\quad R(\text{food}) = 3$ $\quad R(\text{sleep}) = 0$



Question: What is the optimal sequence of states?

$$\text{Walk} + \text{Food} + \text{Sleep} + \ldots = 0 + 3 + 0 + \ldots = 3$$

$$\text{Walk} + \text{Walk} + \ldots + \text{Food} + \text{Sleep} + \ldots = 0 + 0 + \ldots + 3 + 0 + \ldots = 3$$

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Experiment: Place a cookie in front of a child. If they do not eat the cookie for 5 minutes, they get two cookies

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Experiment: Place a cookie in front of a child. If they do not eat the cookie for 5 minutes, they get two cookies

Question: What does the child do?

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Experiment: Place a cookie in front of a child. If they do not eat the cookie for 5 minutes, they get two cookies

Question: What does the child do?

Answer: The child eats the cookie immediately

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Experiment: Place a cookie in front of a child. If they do not eat the cookie for 5 minutes, they get two cookies

Question: What does the child do?

Answer: The child eats the cookie immediately

Humans and animals prefer reward now instead of later

Markov Decision Processes

The return is an infinite sum

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

We can eat food now, or in 1000 years, the return is the same

Experiment: Place a cookie in front of a child. If they do not eat the cookie for 5 minutes, they get two cookies

Question: What does the child do?

Answer: The child eats the cookie immediately

Humans and animals prefer reward now instead of later

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

What if we make future rewards less important?

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

What if we make future rewards less important?

$$R(s) = \{1 \mid s \in S\}$$

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

What if we make future rewards less important?

$$R(s) = \{1 \mid s \in S\}$$

$$G = \sum_{t=0}^{\infty} 1 = 1 + 1 + 1 + \dots$$

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

What if we make future rewards less important?

$$R(s) = \{1 \mid s \in S\}$$

$$G = \sum_{t=0}^{\infty} 1 = 1 + 1 + \dots$$

$$G = \quad ? \quad = 1 + 0.9 + 0.8 + \dots$$

Markov Decision Processes

$$G = \sum_{t=0}^{\infty} R(s_{t+1})$$

Question: How can we fix the return to prefer rewards sooner?

What if we make future rewards less important?

$$R(s) = \{1 \mid s \in S\}$$

$$G = \sum_{t=0}^{\infty} 1 = 1 + 1 + \dots$$

$$G = \quad ? \quad = 1 + 0.9 + 0.8 + \dots$$

Question: How?

Markov Decision Processes

We can introduce a **discount** term $\gamma \in [0, 1]$ to the return

With $\gamma = 1$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Markov Decision Processes

We can introduce a **discount** term $\gamma \in [0, 1]$ to the return

With $\gamma = 1$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

$$G = 1 + 1 + 1 + \ldots$$

With $\gamma = 0.9$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Markov Decision Processes

We can introduce a **discount** term $\gamma \in [0, 1]$ to the return

With $\gamma = 1$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

$$G = 1 + 1 + 1 + \ldots$$

With $\gamma = 0.9$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

$$G = (0.9^0 \cdot 1) + (0.9^1 \cdot 1) + (0.9^2 \cdot 1) + \ldots$$

Markov Decision Processes

We can introduce a **discount** term $\gamma \in [0, 1]$ to the return

With $\gamma = 1$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

$$G = 1 + 1 + 1 + \dots$$

With $\gamma = 0.9$

$$G = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

$$G = (0.9^0 \cdot 1) + (0.9^1 \cdot 1) + (0.9^2 \cdot 1) + \dots$$

$$G = 1 + 0.9 + 0.81 + \dots$$

Markov Decision Processes

Without γ

$$G = 1 + 1 + 1 + ...$$

With γ

$$G = (0.9^0 \cdot 1) + (0.9^1 \cdot 1) + (0.9^2 \cdot 1) + ...$$

$$G = 1 + 0.9 + 0.81 + ...$$

We call this the **discounted return**

Markov Decision Processes

Without γ

$$G = 1 + 1 + 1 + \ldots$$

With γ

$$G = (0.9^0 \cdot 1) + (0.9^1 \cdot 1) + (0.9^2 \cdot 1) + \ldots$$

$$G = 1 + 0.9 + 0.81 + \ldots$$

We call this the **discounted return**

Thus, our objective is

$$\arg \max_{s \in S} G = \arg \max_{s \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Markov Decision Processes

Let us review

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space
- A is the action space

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space
- A is the action space
- $T : S \times A \mapsto \Delta S$ is the state transition function

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space
- A is the action space
- $T : S \times A \mapsto \Delta S$ is the state transition function
- $R : S \mapsto \mathbb{R}$ is the reward function

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space
- A is the action space
- $T : S \times A \mapsto \Delta S$ is the state transition function
- $R : S \mapsto \mathbb{R}$ is the reward function
- $\gamma \in [0, 1]$ is the discount factor

Markov Decision Processes

Let us review

Definition: A Markov decision process (MDP) is a tuple (S, A, T, R, γ)

- S is the state space
- A is the action space
- $T : S \times A \mapsto \Delta S$ is the state transition function
- $R : S \mapsto \mathbb{R}$ is the reward function
- $\gamma \in [0, 1]$ is the discount factor

For the rest of the course, we will solve MDPs

Markov Decision Processes

Reinforcement learning is designed to solve MDPs

Markov Decision Processes

Reinforcement learning is designed to solve MDPs

In reinforcement learning, we have a single goal

Markov Decision Processes

Reinforcement learning is designed to solve MDPs

In reinforcement learning, we have a single goal

Maximize the discounted return

Markov Decision Processes

Reinforcement learning is designed to solve MDPs

In reinforcement learning, we have a single goal

Maximize the discounted return

$$\arg \max_{s \in S} G = \arg \max_{s \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Markov Decision Processes

Reinforcement learning is designed to solve MDPs

In reinforcement learning, we have a single goal

Maximize the discounted return

$$\underset{s \in S}{\arg\max}\, G = \underset{s \in S}{\arg\max} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

You must understand the discounted return!

Markov Decision Processes

Understanding MDPs is the **most important part** of RL

Markov Decision Processes

Understanding MDPs is the **most important part** of RL

Existing software can train RL agents on your MDP

Markov Decision Processes

Understanding MDPs is the **most important part** of RL

Existing software can train RL agents on your MDP

You can train an RL agent without understanding RL

Markov Decision Processes

Understanding MDPs is the **most important part** of RL

Existing software can train RL agents on your MDP

You can train an RL agent without understanding RL

You can only train an agent if you can model your problem as an MDP

Markov Decision Processes

Understanding MDPs is the **most important part** of RL

Existing software can train RL agents on your MDP

You can train an RL agent without understanding RL

You can only train an agent if you can model your problem as an MDP

Make sure you understand MDPs!

Exercise

Exercise

Model Super Mario Bros as an MDP

Exercise

Model Super Mario Bros as an MDP Design the:

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R
- Discount factor γ

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R
- Discount factor γ

Compute discounted return for:

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R
- Discount factor γ

Compute discounted return for:

- Eat mushroom at $t = 10$

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R
- Discount factor γ

Compute discounted return for:

- Eat mushroom at $t = 10$
- Collect coins at $t = 11, 12$

Exercise

Model Super Mario Bros as an MDP Design the:

- State space S
- Action space A
- State transition function T
- Reward function R
- Discount factor γ

Compute discounted return for:

- Eat mushroom at $t = 10$
- Collect coins at $t = 11, 12$
- Die to bowser at $t = 20$

Coding

Coding

In this course, we will implemented MDPs using **gymnasium**

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

- state space (S)

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

- state space (S)
- action space (A)

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

- state space (S)
- action space (A)
- step (T, R, terminated)

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

- state space (S)
- action space (A)
- step (T, R, terminated)
- reset (s_0)

Coding

In this course, we will implemented MDPs using **gymnasium**

Originally developed by OpenAI for reinforcement learning

Gymnasium provides an **environment** (MDP) API

Must define:

- state space (S)
- action space (A)
- step (T, R, terminated)
- reset (s_0)

https://gymnasium.farama.org/api/env/

Coding

Gymnasium uses **observations** instead of **states**

Coding

Gymnasium uses **observations** instead of **states**

Question: What was the condition for MDPs?

Coding

Gymnasium uses **observations** instead of **states**

Question: What was the condition for MDPs?

The next Markov state only depends on the current Markov state

Coding

Gymnasium uses **observations** instead of **states**

Question: What was the condition for MDPs?

The next Markov state only depends on the current Markov state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

Coding

Gymnasium uses **observations** instead of **states**

Question: What was the condition for MDPs?

The next Markov state only depends on the current Markov state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

If the Markov property is broken, $s_t \in S$ is not a Markov state

Coding

Gymnasium uses **observations** instead of **states**

Question: What was the condition for MDPs?

The next Markov state only depends on the current Markov state

$$\Pr(s_t \mid s_{t-1}, s_{t-2}, ..., s_1) = \Pr(s_t \mid s_{t-1})$$

If the Markov property is broken, $s_t \in S$ is not a Markov state

Then, we change $s_t \in S$ to an **observation** $o_t \in O$ (more later)

Coding

```
import gymnasium as gym

MyMDP(gym.Env):
  def __init__(self):
    self.action_space = gym.spaces.Discrete(3) # A
    self.observation_space = gym.spaces.Discrete(5) # S

  def reset(self, seed=None) -> Tuple[Observation, Dict]

  def step(self, action) -> Tuple[
    Observation, Reward, Terminated, Truncated, Dict
  ]
```

Coding

https://colab.research.google.com/drive/1rDNik5oRl27si8wdtMLE7Y41U5J2bx-I#scrollTo=9pOLI5OgKvoE

Exam Next Class

Exam Next Class

We will have an exam next week

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

No book, no notes, no calculator – only pencil and pen

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

No book, no notes, no calculator – only pencil and pen

Study **notation**, **probability**, bandits, and MDPs

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

No book, no notes, no calculator – only pencil and pen

Study **notation**, **probability**, bandits, and MDPs

Practice expectations, bandit problems, state transitions, and returns

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

No book, no notes, no calculator – only pencil and pen

Study **notation**, **probability**, bandits, and MDPs

Practice expectations, bandit problems, state transitions, and returns

You must have intuition, not memorize

Exam Next Class

We will have an exam next week

1 hour 15 minutes, no coding, only math

No book, no notes, no calculator – only pencil and pen

Study **notation**, **probability**, bandits, and MDPs

Practice expectations, bandit problems, state transitions, and returns

You must have intuition, not memorize

Too many A's last term, exam will be **difficult**