Algorithms

CISC 7404 - Decision Making

Steven Morad

University of Macau

Quiz results on moodle

Quiz results on moodle

If you have no score, come see me

Quiz results on moodle

If you have no score, come see me

Mean score is $\frac{3.37}{4} \approx 84\%$

Quiz results on moodle

If you have no score, come see me

Mean score is $\frac{3.37}{4} \approx 84\%$

You did better than expected!

Quiz results on moodle

If you have no score, come see me

Mean score is $\frac{3.37}{4} \approx 84\%$

You did better than expected!

If mean course score is $> 80\%$ but you understand the material it is ok

Quiz results on moodle

If you have no score, come see me

Mean score is $\frac{3.37}{4} \approx 84\%$

You did better than expected!

If mean course score is $> 80\%$ but you understand the material it is ok

I will not decrease total score

Quiz results on moodle

If you have no score, come see me

Mean score is $\frac{3.37}{4} \approx 84\%$

You did better than expected!

If mean course score is $> 80\%$ but you understand the material it is ok

I will not decrease total score

Do not forget individual participation grade!

Review

Review

Diffusion models

Review

Diffusion models

https://arxiv.org/pdf/2006.11239

Algorithms

Algorithms

Algorithms

Our goal is to maximize the discounted return

Algorithms

Our goal is to maximize the discounted return

Take actions in the MDP to maximize the discounted return

Algorithms

Our goal is to maximize the discounted return

Take actions in the MDP to maximize the discounted return

We introduce a **policy** to select actions

Algorithms

Our goal is to maximize the discounted return

Take actions in the MDP to maximize the discounted return

We introduce a **policy** to select actions

$$\pi : S \times \Theta \mapsto \Delta A$$

Algorithms

Our goal is to maximize the discounted return

Take actions in the MDP to maximize the discounted return

We introduce a **policy** to select actions

$$\pi : S \times \Theta \mapsto \Delta A$$

The policy is the "brain" of the agent

Algorithms

Our goal is to maximize the discounted return

Take actions in the MDP to maximize the discounted return

We introduce a **policy** to select actions

$$\pi : S \times \Theta \mapsto \Delta A$$

The policy is the "brain" of the agent

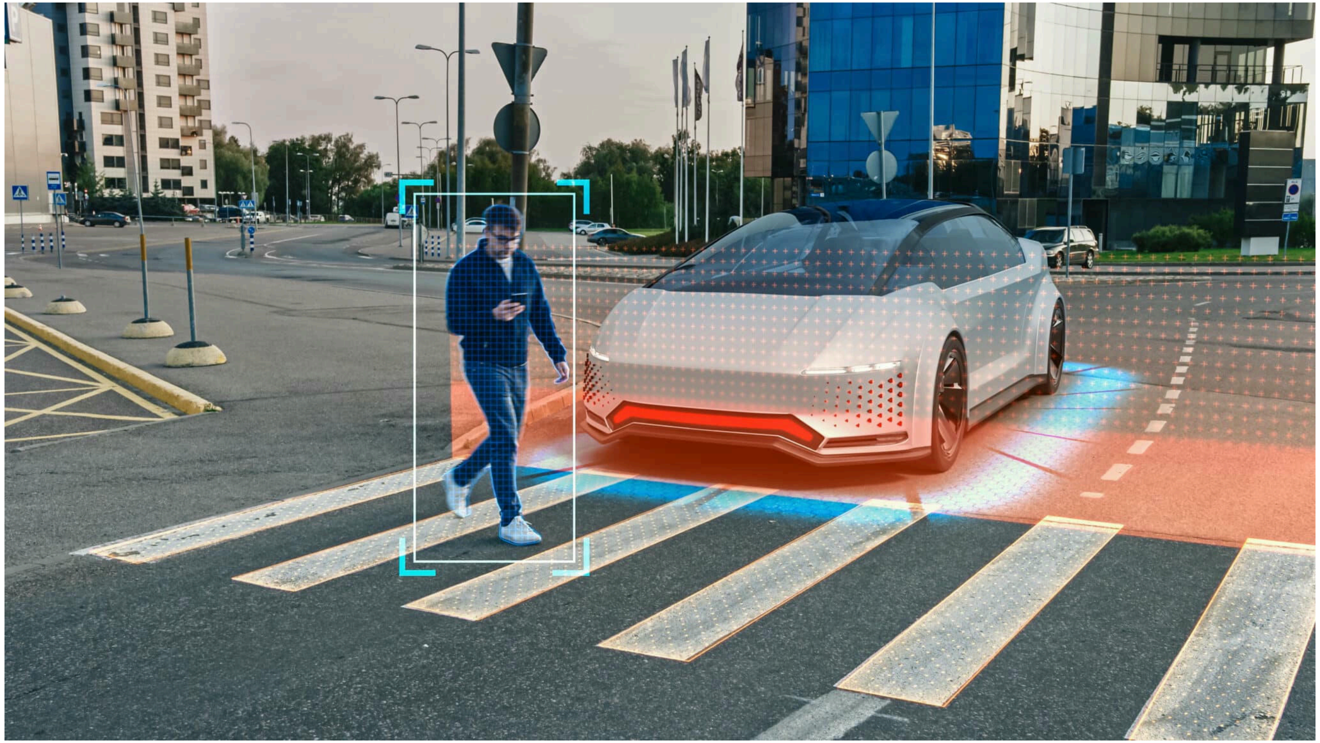
It makes decisions for the agent

Algorithms

Policies can be good, bad, or even human!

Algorithms

Policies can be good, bad, or even human!



Algorithms

We use **algorithms** to find good policies

Algorithms

We use **algorithms** to find good policies

Question: What makes a policy good?

Algorithms

We use **algorithms** to find good policies

Question: What makes a policy good?

Answer: It achieves a large discounted return

Algorithms

We use **algorithms** to find good policies

Question: What makes a policy good?

Answer: It achieves a large discounted return

Almost all the algorithms we learn in this course have guarantees

Algorithms

We use **algorithms** to find good policies

Question: What makes a policy good?

Answer: It achieves a large discounted return

Almost all the algorithms we learn in this course have guarantees

That is, if you train long enough, your policy will become optimal

Algorithms

We use **algorithms** to find good policies

Question: What makes a policy good?

Answer: It achieves a large discounted return

Almost all the algorithms we learn in this course have guarantees

That is, if you train long enough, your policy will become optimal

The policy is guaranteed to maximize the discounted return

Algorithms

Today, we will derive the **trajectory optimization** algorithm

Algorithms

Today, we will derive the **trajectory optimization** algorithm

This algorithm is old, and does not require deep learning

Algorithms

Today, we will derive the **trajectory optimization** algorithm

This algorithm is old, and does not require deep learning

These ideas appear in classical robotics and control theory

Algorithms

Today, we will derive the **trajectory optimization** algorithm

This algorithm is old, and does not require deep learning

These ideas appear in classical robotics and control theory

https://www.youtube.com/watch?v=6qj3EfRTtkE

Algorithms

There are two classes of algorithms

Algorithms

There are two classes of algorithms

Model-based

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

We do not know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

We do not know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Expensive to train, cheap to use

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

We do not know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Expensive to train, cheap to use

Closer to deep learning

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

We do not know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Expensive to train, cheap to use

Closer to deep learning

Today, we will cover a model-based algorithm called trajectory optimization

Algorithms

There are two classes of algorithms

Model-based

We know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cheap to train, expensive to use

Closer to traditional control theory

Model-free

We do not know $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Expensive to train, cheap to use

Closer to deep learning

Today, we will cover a model-based algorithm called trajectory optimization

Critical part of Alpha-* methods (AlphaGo, AlphaStar, AlphaZero)

Algorithms

Recall the discounted return, our objective for the rest of this course

Algorithms

Recall the discounted return, our objective for the rest of this course

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

Algorithms

Recall the discounted return, our objective for the rest of this course

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

$$\tau = \begin{bmatrix} s_0 & a_0 \\ s_1 & a_1 \\ \vdots & \vdots \end{bmatrix}$$

Algorithms

Recall the discounted return, our objective for the rest of this course

$$G(\boldsymbol{\tau}) = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1}) \qquad \tau = \begin{bmatrix} s_0 & a_0 \\ s_1 & a_1 \\ \vdots & \vdots \end{bmatrix}$$

We want to maximize the discounted return

$$\arg \max_{\boldsymbol{\tau}} G(\boldsymbol{\tau}) = \arg \max_{s_1, s_2, \ldots \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Algorithms

Recall the discounted return, our objective for the rest of this course

$$G(\boldsymbol{\tau}) = \sum_{t=0}^{\infty} \gamma^t R(s_{t+1}) \qquad \tau = \begin{bmatrix} s_0 & a_0 \\ s_1 & a_1 \\ \vdots & \vdots \end{bmatrix}$$

We want to maximize the discounted return

$$\arg\max_{\boldsymbol{\tau}} G(\boldsymbol{\tau}) = \arg\max_{s_1, s_2, \ldots \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

We want to find τ that provides the greatest discounted return

Algorithms

$$\arg\max_{\boldsymbol{\tau}} G(\boldsymbol{\tau}) = \arg\max_{s_1, s_2, \ldots \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

Algorithms

$$\arg\max_{\boldsymbol{\tau}} G(\boldsymbol{\tau}) = \arg\max_{s_1, s_2, \ldots \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

This objective looks simple, but $R(s_{t+1})$ hides much of the process

Algorithms

$$\arg \max_{\boldsymbol{\tau}} G(\boldsymbol{\tau}) = \arg \max_{s_1, s_2, \ldots \in S} \sum_{t=0}^{\infty} \gamma^t R(s_{t+1})$$

This objective looks simple, but $R(s_{t+1})$ hides much of the process

To understand what is hiding, let us examine the reward function

Reward Optimization

Reward Optimization

Consider the reward function

$$R(s_{t+1})$$

Reward Optimization

Consider the reward function

$$R(s_{t+1})$$

Perhaps we want to maximize the reward

$$\arg\max_{s_{t+1} \in S} R(s_{t+1})$$

Reward Optimization

Consider the reward function

$$R(s_{t+1})$$

Perhaps we want to maximize the reward

$$\arg\max_{s_{t+1} \in S} R(s_{t+1})$$

Question: In state s_t, take action a_t, what is $R(s_{t+1})$?

Reward Optimization

Consider the reward function

$$R(s_{t+1})$$

Perhaps we want to maximize the reward

$$\arg\max_{s_{t+1} \in S} R(s_{t+1})$$

Question: In state s_t, take action a_t, what is $R(s_{t+1})$?

Answer: Not sure. $R(s_{t+1})$ depends on $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Reward Optimization

Consider the reward function

$$R(s_{t+1})$$

Perhaps we want to maximize the reward

$$\arg\max_{s_{t+1} \in S} R(s_{t+1})$$

Question: In state s_t, take action a_t, what is $R(s_{t+1})$?

Answer: Not sure. $R(s_{t+1})$ depends on $\mathrm{Tr}(s_{t+1} \mid s_t, a_t)$

Cannot know s_{t+1} with certainty, only know the distribution!

Reward Optimization

s_{t+1} is the **outcome** of a random process

Reward Optimization

s_{t+1} is the **outcome** of a random process

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Reward Optimization

s_{t+1} is the **outcome** of a random process

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Question: What is S?

Reward Optimization

s_{t+1} is the **outcome** of a random process

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Question: What is S?

Answer: State space, also the outcome space Ω of Tr

Reward Optimization

s_{t+1} is the **outcome** of a random process

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Question: What is S?

Answer: State space, also the outcome space Ω of Tr

$$s_{t+1} \in S \equiv \omega \in \Omega$$

Reward Optimization

s_{t+1} is the **outcome** of a random process

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Question: What is S?

Answer: State space, also the outcome space Ω of Tr

$$s_{t+1} \in S \equiv \omega \in \Omega$$

Question: Ok, now what is the definition of R?

Answer:

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

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Answer: R is a random variable!

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Answer: R is a random variable!

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

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Answer: R is a random variable!

$R : S \mapsto \mathbb{R}$ $\qquad$ $S = \Omega$

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Answer: R is a random variable!

$R : S \mapsto \mathbb{R}$ $\qquad$ $S = \Omega$ $\qquad$ $R : \Omega \mapsto \mathbb{R}$

Reward Optimization

$$s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t), \quad s_t, s_{t+1} \in S$$

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

If you can answer the following question, you understand the course

Question: R is a special kind of function, what is it?

Answer: R is a random variable!

$R : S \mapsto \mathbb{R}$ $\qquad$ $S = \Omega$ $\qquad$ $R : \Omega \mapsto \mathbb{R}$

We should write it as $\mathcal{R} : S \mapsto \mathbb{R}$

Reward Optimization

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

Reward Optimization

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

Question: What do we like to do with random variables?

Reward Optimization

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

Question: What do we like to do with random variables?

Answer: Take the expectation!

Reward Optimization

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

Question: What do we like to do with random variables?

Answer: Take the expectation!

Question: Why do we like to take the expectation of random variables?

Reward Optimization

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

Question: What do we like to do with random variables?

Answer: Take the expectation!

Question: Why do we like to take the expectation of random variables?

Answer: It maps random processes to something we can maximize

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

We cannot know for certain which reward we get in the next timestep

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

We cannot know for certain which reward we get in the next timestep

But we can know the **average** reward using the expectation

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

We cannot know for certain which reward we get in the next timestep

But we can know the **average** reward using the expectation

$$\mathbb{E}[\mathcal{X}] = \sum_{\omega \in \Omega} \mathcal{X}(\omega) \cdot \Pr(\omega)$$

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

We cannot know for certain which reward we get in the next timestep

But we can know the **average** reward using the expectation

$$\mathbb{E}[\mathcal{X}] = \sum_{\omega \in \Omega} \mathcal{X}(\omega) \cdot \Pr(\omega)$$

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\mathcal{R}(s_{t+1}), \quad s_{t+1} \sim \mathrm{Tr}(\cdot \mid s_t, a_t)$$

We cannot know for certain which reward we get in the next timestep

But we can know the **average** reward using the expectation

$$\mathbb{E}[\mathcal{X}] = \sum_{\omega \in \Omega} \mathcal{X}(\omega) \cdot \Pr(\omega)$$

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Random variable conditioned on s_t, a_t

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

As an agent, we have partial control of the future reward

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

As an agent, we have partial control of the future reward

We can choose an action a_t to maximize the expected reward

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

As an agent, we have partial control of the future reward

We can choose an action a_t to maximize the expected reward

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

As an agent, we have partial control of the future reward

We can choose an action a_t to maximize the expected reward

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Note: Cannot directly maximize $\mathcal{R}$ because s_{t+1} is random

Reward Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

As an agent, we have partial control of the future reward

We can choose an action a_t to maximize the expected reward

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Note: Cannot directly maximize $\mathcal{R}$ because s_{t+1} is random

We can maximize the expected reward

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2.
3.
4.

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3.
4.

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3. `expected_reward = [sum([p * rewards]) for p in probs]`
4.

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3. `expected_reward = [sum([p * rewards]) for p in probs]`
4. `a = argmax(expected_reward)`

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3. `expected_reward = [sum([p * rewards]) for p in probs]`
4. `a = argmax(expected_reward)`

Question: Have we seen something similar before?

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3. `expected_reward = [sum([p * rewards]) for p in probs]`
4. `a = argmax(expected_reward)`

Question: Have we seen something similar before?

Answer: Bandits!

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

How to code this:

1. `probs = [[Tr(next_s, s, a) for next_s in S] for a in A]`
2. `rewards = [R(next_s) for next_s in S]`
3. `expected_reward = [sum([p * rewards]) for p in probs]`
4. `a = argmax(expected_reward)`

Question: Have we seen something similar before?

Answer: Bandits!

$$\arg\max_{a \in \{1 \dots k\}} \mathbb{E}[\mathcal{X}_a]$$

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Earlier, we said that algorithms provide a policy $\pi : S \times \Theta \mapsto \Delta A$

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Earlier, we said that algorithms provide a policy $\pi : S \times \Theta \mapsto \Delta A$

But this equation is not yet a policy!

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Earlier, we said that algorithms provide a policy $\pi : S \times \Theta \mapsto \Delta A$

But this equation is not yet a policy!

Let us turn this equation into a policy

Reward Optimization

$$\underset{a_t \in A}{\arg\max}\, \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \underset{a_t \in A}{\arg\max} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

$$\pi : S \times \Theta \mapsto \Delta A$$

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

$$\pi : S \times \Theta \mapsto \Delta A$$

Question: How to make equation into policy? (Hint: Greedy policy)

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

$$\pi : S \times \Theta \mapsto \Delta A$$

Question: How to make equation into policy? (Hint: Greedy policy)

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

$$\pi : S \times \Theta \mapsto \Delta A$$

Question: How to make equation into policy? (Hint: Greedy policy)

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

Policy maximizes the reward at each timestep

Reward Optimization

$$\arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \arg\max_{a_t \in A} \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

$$\pi : S \times \Theta \mapsto \Delta A$$

Question: How to make equation into policy? (Hint: Greedy policy)

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

Policy maximizes the reward at each timestep

$$\underbrace{\pi(a_0 \mid s_0)}_{\mathcal{R}(s_1)}, \underbrace{\pi(a_1 \mid s_1)}_{\mathcal{R}(s_2)}, \ldots$$

Reward Optimization

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

Reward Optimization

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

$$\underbrace{\pi(a_0 \mid s_0)}_{\mathcal{R}(s_1)}, \underbrace{\pi(a_1 \mid s_1)}_{\mathcal{R}(s_2)}, \ldots$$

Reward Optimization

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

$$\underbrace{\pi(a_0 \mid s_0)}_{\mathcal{R}(s_1)}, \underbrace{\pi(a_1 \mid s_1)}_{\mathcal{R}(s_2)}, \ldots$$

This policy is **optimal** with respect to the expected reward

Reward Optimization

$$\pi(a_t \mid s_t; \theta) = \Pr(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

$$\underbrace{\pi(a_0 \mid s_0)}_{\mathcal{R}(s_1)}, \underbrace{\pi(a_1 \mid s_1)}_{\mathcal{R}(s_2)}, \dots$$

This policy is **optimal** with respect to the expected reward

It will always act to maximize the expected reward

Reward Optimization

Example: Online advertising, show users ads so they buy products

Reward Optimization

Example: Online advertising, show users ads so they buy products

$$S = \{0, 1\}^d \times \{0, 1\}$$

Current user info, prev user buy?

Reward Optimization

Example: Online advertising, show users ads so they buy products

$$S = \{0, 1\}^d \times \{0, 1\}$$

Current user info, prev user buy?

$$A = [0, 1]^{256 \times 256 \times 3}$$

Pixels of advertisement image

Reward Optimization

Example: Online advertising, show users ads so they buy products

$$S = \{0, 1\}^d \times \{0, 1\}$$

Current user info, prev user buy?

$$A = [0, 1]^{256 \times 256 \times 3}$$

Pixels of advertisement image

$$\mathcal{R}(s_{t+1}) = \begin{cases} 1 \text{ if bought product} \\ 0 \text{ otherwise} \end{cases}$$

Reward Optimization

Example: Online advertising, show users ads so they buy products

$$S = \{0, 1\}^d \times \{0, 1\}$$

Current user info, prev user buy?

$$A = [0, 1]^{256 \times 256 \times 3}$$

Pixels of advertisement image

$$\mathcal{R}(s_{t+1}) = \begin{cases} 1 \text{ if bought product} \\ 0 \text{ otherwise} \end{cases}$$

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

Reward Optimization

Example: Online advertising, show users ads so they buy products

$$S = \{0, 1\}^d \times \{0, 1\}$$ Current user info, prev user buy?

$$A = [0, 1]^{256 \times 256 \times 3}$$ Pixels of advertisement image

$$\mathcal{R}(s_{t+1}) = \begin{cases} 1 \text{ if bought product} \\ 0 \text{ otherwise} \end{cases}$$

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] \\ 0 \text{ otherwise} \end{cases}$$

This will create the best ad creator possible!

Reward Optimization

Question: Are we done? Is maximizing reward enough?

Reward Optimization

Question: Are we done? Is maximizing reward enough?

Answer: No, maximize the discounted return, not the reward!

Reward Optimization

Question: Are we done? Is maximizing reward enough?

Answer: No, maximize the discounted return, not the reward!

We have one more thing to do

Trajectory Optimization

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

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

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

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

What we want:

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

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

What we want:

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?] = ?$$

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?] = ?$$

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?\,] = ?$$

Note: G is also a random variable

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?] = ?$$

Note: G is also a random variable

$$G : \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \mapsto \mathbb{R}$$

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ? \,] = ?$$

Note: G is also a random variable

$$G : \quad \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \quad \mapsto \mathbb{R}$$

We can rewrite it curly since it is a random variable

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?\,] = ?$$

Note: G is also a random variable

$$G : \quad \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \quad \mapsto \mathbb{R}$$

We can rewrite it curly since it is a random variable

$$\mathcal{G} : \quad \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \quad \mapsto \mathbb{R}$$

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?\,] = ?$$

Note: G is also a random variable

$$G : \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \mapsto \mathbb{R}$$

We can rewrite it curly since it is a random variable

$$\mathcal{G} : \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr}, \pi} \mapsto \mathbb{R}$$

Back to the problem...

Trajectory Optimization

$$\mathbb{E}[G(\boldsymbol{\tau}) \mid ?\,] = ?$$

Note: G is also a random variable

$$G : \quad \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr},\pi} \quad \mapsto \mathbb{R}$$

We can rewrite it curly since it is a random variable

$$\mathcal{G} : \quad \underbrace{S^n \times A^n}_{\text{Outcome of stochastic Tr},\pi} \quad \mapsto \mathbb{R}$$

Back to the problem...

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?\,] = ?$$

Trajectory Optimization

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

Trajectory Optimization

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

Take the expected value of both sides

Trajectory Optimization

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

Take the expected value of both sides

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ?\right]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ? \right]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ? \right]$$

The expectation is a linear function, we can move it inside the sum

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ?\right]$$

The expectation is a linear function, we can move it inside the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \mathbb{E}[\gamma^t \mathcal{R}(s_{t+1}) \mid ?]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ?\right]$$

The expectation is a linear function, we can move it inside the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \mathbb{E}[\gamma^t \mathcal{R}(s_{t+1}) \mid ?]$$

Expectation is linear, can factor out γ

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_{t+1}) \middle| ?\right]$$

The expectation is a linear function, we can move it inside the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \mathbb{E}[\gamma^t \mathcal{R}(s_{t+1}) \mid ?]$$

Expectation is linear, can factor out γ

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?]$$

Now, let's figure out the conditions

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?]$$

Now, let's figure out the conditions

We know $\mathbb{E}[\mathcal{R}(s_1)]$ needs s_0, a_0

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?]$$

Now, let's figure out the conditions

We know $\mathbb{E}[\mathcal{R}(s_1)]$ needs s_0, a_0

$\mathbb{E}[\mathcal{R}(s_2) \mid s_1, a_1]$, already have s_1 only need a_1

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?]$$

Now, let's figure out the conditions

We know $\mathbb{E}[\mathcal{R}(s_1)]$ needs s_0, a_0

$\mathbb{E}[\mathcal{R}(s_2) \mid s_1, a_1]$, already have s_1 only need a_1

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ..., a_t]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?\,] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid ?\,]$$

Now, let's figure out the conditions

We know $\mathbb{E}[\mathcal{R}(s_1)]$ needs s_0, a_0

$\mathbb{E}[\mathcal{R}(s_2) \mid s_1, a_1]$, already have s_1 only need a_1

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid ?\,] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ..., a_t]$$

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ..., a_t]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ...]$$

Unroll the sum

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ...]$$

Unroll the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

$$\gamma^0 \ \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ...]$$

Unroll the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$
$$\gamma^0 \ \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

We know how to compute the first term from before

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, a_1, ...]$$

Unroll the sum

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

$$\gamma^0 \; \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

We know how to compute the first term from before

$$\mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_t, a_t] = \sum_{s_{t+1} \in S} \mathcal{R}(s_{t+1}) \cdot \mathrm{Tr}(s_{t+1} \mid s_t, a_t)$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

$$\gamma^0 \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \ \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

$$\gamma^0 \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

Question: Do we know the second term?

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

$$\gamma^0 \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] + \gamma^1 \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] + \gamma^2 \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] + ...$$

Question: Do we know the second term?

Answer: It is more tricky

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]$$

$\mathcal{R}(s_2)$ needs s_2, but we only have s_0!

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]$$

$\mathcal{R}(s_2)$ needs s_2, but we only have s_0!

For $\mathcal{R}(s_1)$ relies on the distribution $\mathrm{Tr}(s_1 \mid s_0, a_0)$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]$$

$\mathcal{R}(s_2)$ needs s_2, but we only have s_0!

For $\mathcal{R}(s_1)$ relies on the distribution $\mathrm{Tr}(s_1 \mid s_0, a_0)$

For $\mathcal{R}(s_2)$, the reward relies on $\mathrm{Tr}(s_2 \mid s_1, a_1)$ and $\mathrm{Tr}(s_1 \mid s_0, a_0)$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]$$

$\mathcal{R}(s_2)$ needs s_2, but we only have s_0!

For $\mathcal{R}(s_1)$ relies on the distribution $\mathrm{Tr}(s_1 \mid s_0, a_0)$

For $\mathcal{R}(s_2)$, the reward relies on $\mathrm{Tr}(s_2 \mid s_1, a_1)$ and $\mathrm{Tr}(s_1 \mid s_0, a_0)$

For $\mathcal{R}(s_{n+1})$ we need an expression for $\Pr(s_{n+1} \mid s_0, a_0, a_1, \ldots)$

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Answer: In lecture 3 we found the probability of a future state in a Markov process

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Answer: In lecture 3 we found the probability of a future state in a Markov process

$$\Pr(s_{n+1} \mid s_0) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t)$$

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Answer: In lecture 3 we found the probability of a future state in a Markov process

$$\Pr(s_{n+1} \mid s_0) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t)$$

To extend to MDP, just need to include the actions!

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Answer: In lecture 3 we found the probability of a future state in a Markov process

$$\Pr(s_{n+1} \mid s_0) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t)$$

To extend to MDP, just need to include the actions!

$$\Pr(s_{n+1} \mid s_0, a_0, a_1, ..., a_{n-1}) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)$$

Trajectory Optimization

Question: How do we find $\Pr(s_{n+1} \mid s_0, a_0, a_1, ...)$?

Answer: In lecture 3 we found the probability of a future state in a Markov process

$$\Pr(s_{n+1} \mid s_0) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t)$$

To extend to MDP, just need to include the actions!

$$\Pr(s_{n+1} \mid s_0, a_0, a_1, ..., a_{n-1}) = \sum_{s_1, s_2, ... s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)$$

This predicts the future states of an MDP

Trajectory Optimization

Combine s_{n+1} distribution with $\mathcal{R}$ to predict future rewards

Trajectory Optimization

Combine s_{n+1} distribution with $\mathcal{R}$ to predict future rewards

$$\mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] = \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0)$$

Trajectory Optimization

Combine s_{n+1} distribution with $\mathcal{R}$ to predict future rewards

$$\mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] = \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0)$$

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] = \sum_{s_2 \in S} \mathcal{R}(s_2) \sum_{s_1 \in S} \operatorname{Tr}(s_2 \mid s_1, a_1) \operatorname{Tr}(s_1 \mid s_0, a_0)$$

Trajectory Optimization

Combine s_{n+1} distribution with $\mathcal{R}$ to predict future rewards

$$\mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] = \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0)$$

$$\mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] = \sum_{s_2 \in S} \mathcal{R}(s_2) \sum_{s_1 \in S} \operatorname{Tr}(s_2 \mid s_1, a_1) \operatorname{Tr}(s_1 \mid s_0, a_0)$$

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \sum_{s_{n+1} \in S} \mathcal{R}(s_{n+1}) \sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \operatorname{Tr}(s_{t+1} \mid s_t, a_t)$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \sum_{s_{n+1} \in S} R(s_{n+1}) \sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \sum_{s_{n+1} \in S} R(s_{n+1}) \sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)$$

What does each piece mean?

Trajectory Optimization

s_{n+1} Distribution

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \sum_{s_{n+1} \in S} R(s_{n+1}) \underbrace{\sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)}_{s_{n+1} \text{ Distribution}}$$

What does each piece mean?

Trajectory Optimization

s_{n+1} Distribution

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \sum_{s_{n+1} \in S} R(s_{n+1}) \sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)$$

Mean reward over possible s_{n+1}

What does each piece mean?

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \underbrace{\sum_{s_{n+1} \in S} R(s_{n+1})}_{\text{Mean reward over possible } s_{n+1}} \overbrace{\sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)}^{s_{n+1} \text{ Distribution}}$$

What does each piece mean?

This is only for a single reward, must plug back into discounted return

Trajectory Optimization

$$\mathbb{E}[\mathcal{R}(s_{n+1}) \mid s_0, a_0, a_1, ..., a_n] = \underbrace{\sum_{s_{n+1} \in S} R(s_{n+1})}_{\text{Mean reward over possible } s_{n+1}} \overbrace{\sum_{s_1, ..., s_n \in S} \prod_{t=0}^{n} \Pr(s_{t+1} \mid s_t, a_t)}^{s_{n+1} \text{ Distribution}}$$

What does each piece mean?

This is only for a single reward, must plug back into discounted return

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, ..., a_t]$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] =$$

Trajectory Optimization

$$\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = \qquad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0]$$

Trajectory Optimization

$$\begin{aligned}\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\ +\gamma & \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1]\end{aligned}$$

Trajectory Optimization

$$\begin{aligned}\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\ +\gamma & \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] \\ +\gamma^2 & \quad \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2]\end{aligned}$$

Trajectory Optimization

$$\begin{aligned}\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\ & + \gamma \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] \\ & + \gamma^2 \quad \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] \\ & + \quad \ldots\end{aligned}$$

Trajectory Optimization

$$\begin{aligned}
\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\
+\gamma & \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] \\
+\gamma^2 & \quad \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] \\
+ & \quad \ldots \\
= & \quad \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0)
\end{aligned}$$

Trajectory Optimization

$$\begin{aligned}
\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\
+\gamma & \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] \\
+\gamma^2 & \quad \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] \\
+ & \quad ... \\
= & \quad \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0) \\
+\gamma & \quad \sum_{s_2 \in S} \mathcal{R}(s_2) \sum_{s_1 \in S} \operatorname{Tr}(s_2 \mid s_1, a_1) \operatorname{Tr}(s_1 \mid s_0, a_0)
\end{aligned}$$

Trajectory Optimization

$$\begin{aligned}
\mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, ...] = & \quad \mathbb{E}[\mathcal{R}(s_1) \mid s_0, a_0] \\
+\gamma & \quad \mathbb{E}[\mathcal{R}(s_2) \mid s_0, a_0, a_1] \\
+\gamma^2 & \quad \mathbb{E}[\mathcal{R}(s_3) \mid s_0, a_0, a_1, a_2] \\
+ & \quad ... \\
= & \quad \sum_{s_1 \in S} \mathcal{R}(s_1) \operatorname{Tr}(s_1 \mid s_0, a_0) \\
+\gamma & \quad \sum_{s_2 \in S} \mathcal{R}(s_2) \sum_{s_1 \in S} \operatorname{Tr}(s_2 \mid s_1, a_1) \operatorname{Tr}(s_1 \mid s_0, a_0) \\
+\gamma^2 & \quad \sum_{s_3 \in S} \mathcal{R}(s_3) \sum_{s_2 \in S} \operatorname{Tr}(s_3 \mid s_2, a_2) \sum_{s_1 \in S} \operatorname{Tr}(s_2 \mid s_1, a_1) ... \\
+ & \quad ...
\end{aligned}$$

Trajectory Optimization

To maximize the return, we take the $\arg\max$ over possible actions

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\arg\max_{a_0, a_1, \ldots} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\arg\max_{a_0, a_1, \ldots} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

And turn it into a policy

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\underset{a_0, a_1, \ldots}{\arg\max}\, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

And turn it into a policy

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] \\ 0 \text{ otherwise} \end{cases}$$

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\underset{a_0, a_1, \ldots}{\arg\max} \, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

And turn it into a policy

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] \\ 0 \text{ otherwise} \end{cases}$$

We have a name for this policy in control theory

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\underset{a_0, a_1, \ldots}{\arg\max} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

And turn it into a policy

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] \\ 0 \text{ otherwise} \end{cases}$$

We have a name for this policy in control theory

Question: Anyone know what we call it?

Trajectory Optimization

To maximize the return, we take the arg max over possible actions

$$\underset{a_0, a_1, \ldots}{\arg\max} \, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots]$$

And turn it into a policy

$$\pi(a_t \mid s_t; \theta) = \begin{cases} 1 \text{ if } a_t = \arg\max_{a_t \in A} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] \\ 0 \text{ otherwise} \end{cases}$$

We have a name for this policy in control theory

Question: Anyone know what we call it?

Answer: Model Predictive Control (MPC) or Receding Horizon Control

Trajectory Optimization

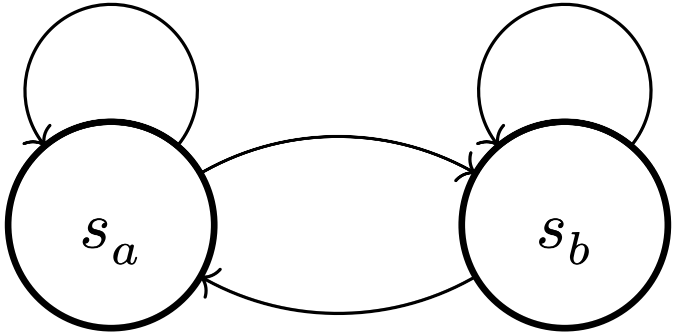
There is a lot of math behind trajectory optimization/MPC

Trajectory Optimization

There is a lot of math behind trajectory optimization/MPC

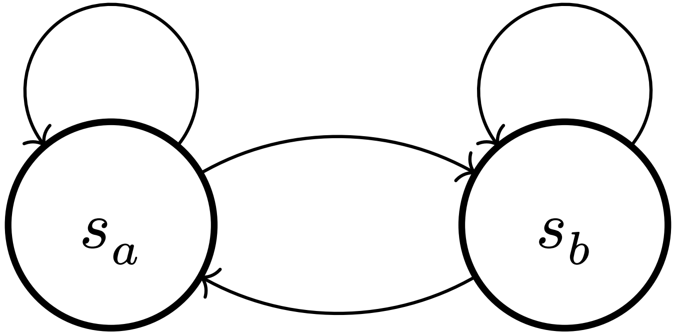
Let us do a visual example to help you understand

Trajectory Optimization



$\mathcal{R}(s_a) = 0$ $\mathcal{R}(s_b) = 1$

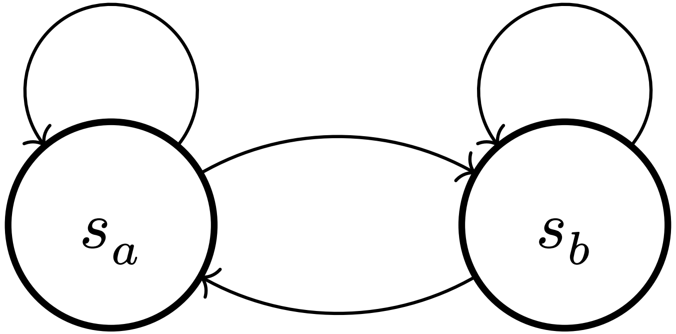
Trajectory Optimization



$\mathcal{R}(s_a) = 0 \qquad \mathcal{R}(s_b) = 1$

$$S = \{s_a, s_b\} \quad A = \{a_a, a_b\}$$

Trajectory Optimization

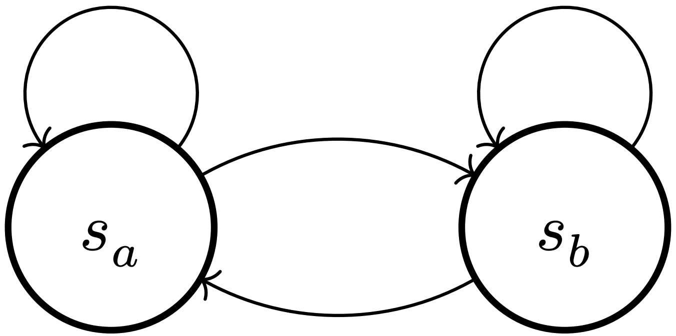


$\mathcal{R}(s_a) = 0$ $\mathcal{R}(s_b) = 1$

$$S = \{s_a, s_b\} \quad A = \{a_a, a_b\}$$

$$\Pr(s_a \mid s_a, a_a) = 0.8; \;\; \Pr(s_b \mid s_a, a_a) = 0.2$$

Trajectory Optimization



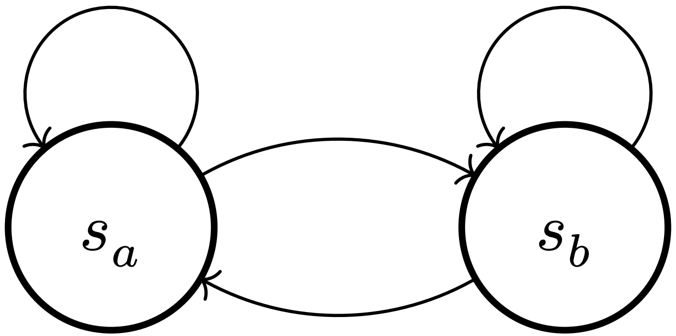
$\mathcal{R}(s_a) = 0 \qquad \mathcal{R}(s_b) = 1$

$$S = \{s_a, s_b\} \quad A = \{a_a, a_b\}$$

$$\Pr(s_a \mid s_a, a_a) = 0.8; \;\; \Pr(s_b \mid s_a, a_a) = 0.2$$

$$\Pr(s_a \mid s_a, a_b) = 0.7; \;\; \Pr(s_b \mid s_a, a_b) = 0.3$$

Trajectory Optimization



$\mathcal{R}(s_a) = 0$ $\mathcal{R}(s_b) = 1$

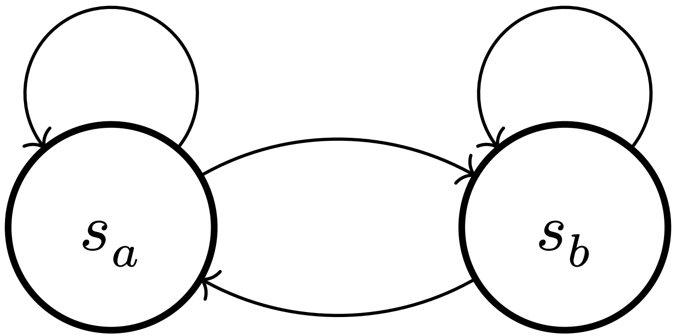
$$S = \{s_a, s_b\} \quad A = \{a_a, a_b\}$$

$$\Pr(s_a \mid s_a, a_a) = 0.8; \ \Pr(s_b \mid s_a, a_a) = 0.2$$

$$\Pr(s_a \mid s_a, a_b) = 0.7; \ \Pr(s_b \mid s_a, a_b) = 0.3$$

$$\Pr(s_a \mid s_b, a_a) = 0.6; \ \Pr(s_b \mid s_a, a_a) = 0.4$$

Trajectory Optimization



$\mathcal{R}(s_a) = 0$ $\quad$ $\mathcal{R}(s_b) = 1$

$$S = \{s_a, s_b\} \quad A = \{a_a, a_b\}$$

$$\Pr(s_a \mid s_a, a_a) = 0.8; \ \Pr(s_b \mid s_a, a_a) = 0.2$$
$$\Pr(s_a \mid s_a, a_b) = 0.7; \ \Pr(s_b \mid s_a, a_b) = 0.3$$

$$\Pr(s_a \mid s_b, a_a) = 0.6; \ \Pr(s_b \mid s_a, a_a) = 0.4$$
$$\Pr(s_a \mid s_b, a_b) = 0.1; \ \Pr(s_b \mid s_a, a_b) = 0.9$$

Trajectory Optimization

We can build this into a decision tree using trajectory optimization

Trajectory Optimization

We can build this into a decision tree using trajectory optimization

The root node of the tree corresponds to s_0

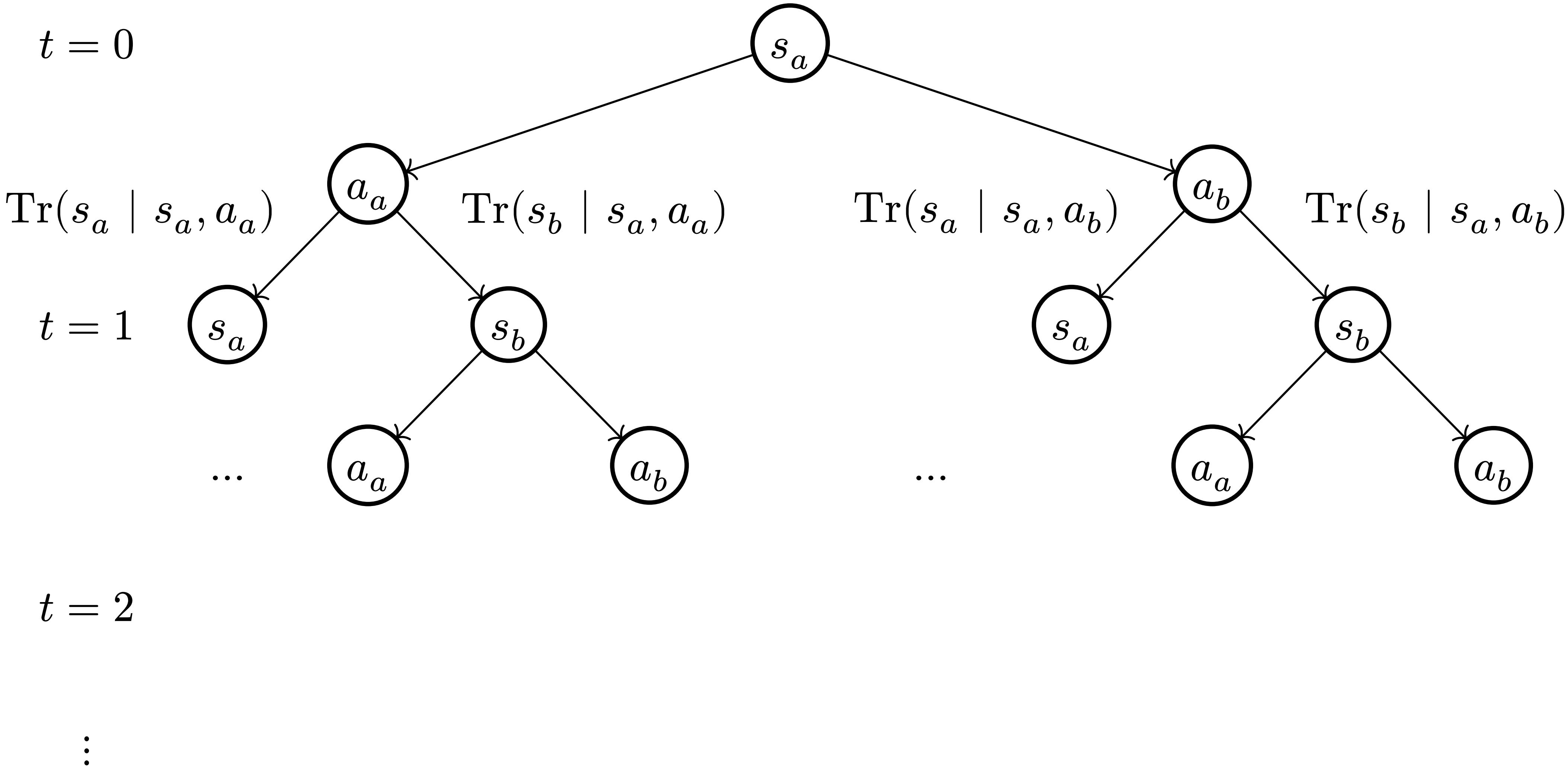
Trajectory Optimization

We can build this into a decision tree using trajectory optimization

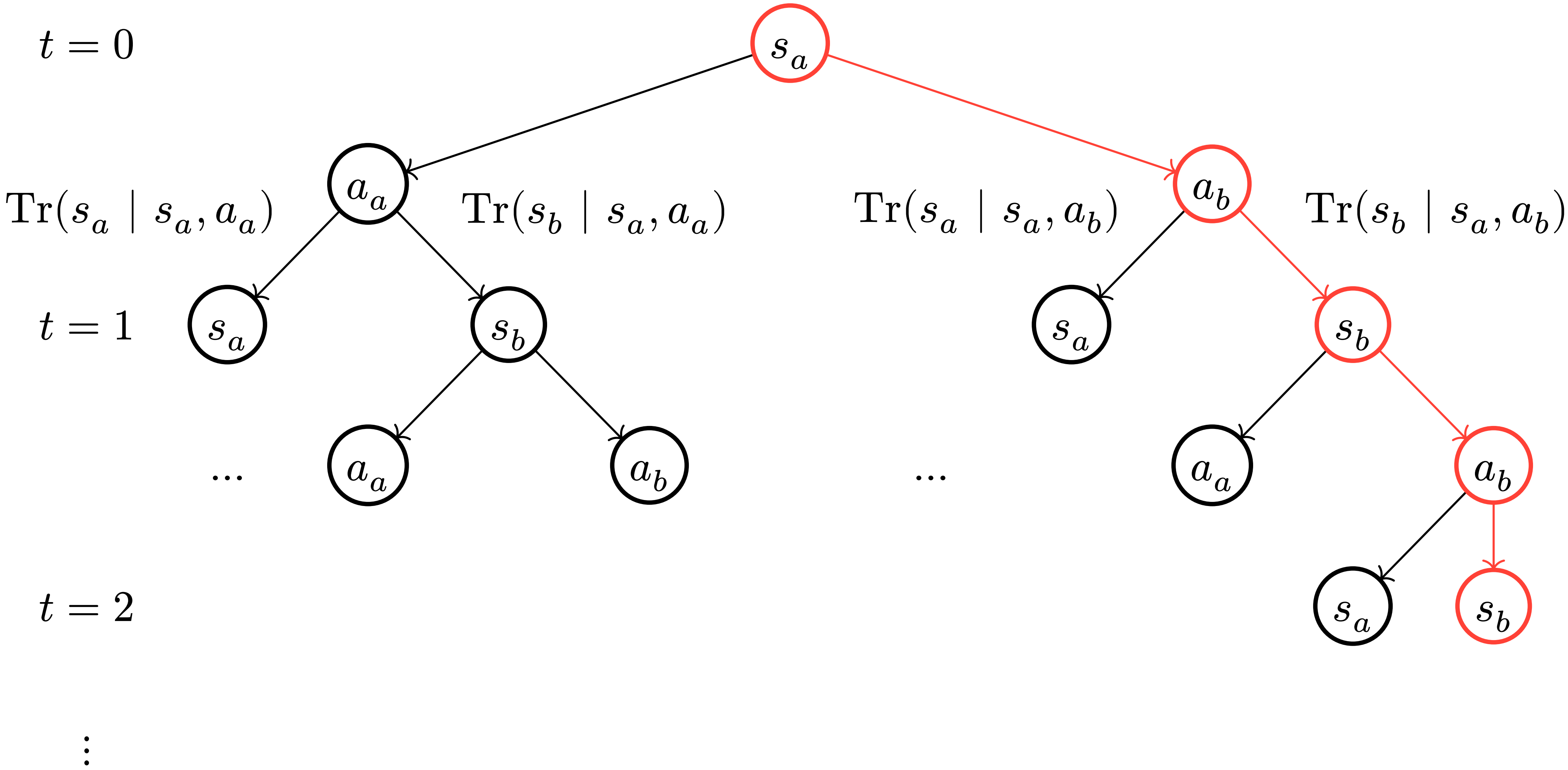
The root node of the tree corresponds to s_0

Each level of the tree enumerates possible outcomes

Trajectory Optimization



Trajectory Optimization



Trajectory Optimization

Question: How many nodes does our tree have?

Trajectory Optimization

Question: How many nodes does our tree have?

Answer: $O(|S| \cdot |A|)^{\infty}$

Trajectory Optimization

Question: How many nodes does our tree have?

Answer: $O(|S| \cdot |A|)^{\infty}$

Question: What does this mean?

Trajectory Optimization

Question: How many nodes does our tree have?

Answer: $O(|S| \cdot |A|)^\infty$

Question: What does this mean?

Answer: Do not have the memory/compute to evaluate all possibilities

Trajectory Optimization

Question: How many nodes does our tree have?

Answer: $O(|S| \cdot |A|)^{\infty}$

Question: What does this mean?

Answer: Do not have the memory/compute to evaluate all possibilities

We have some tricks to make this tractable

Trajectory Optimization

$$\underset{a_0, a_1, \ldots \in A}{\arg\ \max}\ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = \underset{a_0, a_1, \ldots}{\arg\ \max} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Trick 1: Introduce a **horizon** n

Trajectory Optimization

$$\underset{a_0, a_1, \ldots \in A}{\arg\max}\, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = \underset{a_0, a_1, \ldots}{\arg\max} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Trick 1: Introduce a **horizon** n

$$\underset{a_0, \ldots, a_n \in A}{\arg\max}\, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0, \ldots, a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Now we can limit computation to $O(|S| \cdot |A|)^n$

Trajectory Optimization

$$\underset{a_0, a_1, \ldots \in A}{\arg\max} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = \underset{a_0, a_1, \ldots}{\arg\max} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Trick 1: Introduce a **horizon** n

$$\underset{a_0, \ldots, a_n \in A}{\arg\max} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0, \ldots, a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Now we can limit computation to $O(|S| \cdot |A|)^n$

Question: Drawback?

Trajectory Optimization

$$\underset{a_0, a_1, \ldots \in A}{\arg\max}\, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}) \mid s_0, a_0, a_1, \ldots] = \underset{a_0, a_1, \ldots}{\arg\max} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Trick 1: Introduce a **horizon** n

$$\underset{a_0, \ldots, a_n \in A}{\arg\max}\, \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0, \ldots, a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_t]$$

Now we can limit computation to $O(|S| \cdot |A|)^n$

Question: Drawback?

Answer: We no longer consider the infinite future, our agent may get greedy and be trapped

Trajectory Optimization

$$\underset{a_0, ..., a_n \in A}{\arg\max}\ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, ..., a_n] = \underset{a_0, ..., a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, ..., a_n]$$

Trick 2: Only simulate j actions and k states

Trajectory Optimization

$$\underset{a_0, \ldots, a_n \in A}{\arg\max} \ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0, \ldots, a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_n]$$

Trick 2: Only simulate j actions and k states

$$\underset{a_0, \ldots, a_n \sim A^j}{\arg\max} \ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0, \ldots, a_n \sim A^j}{\arg\max} \sum_{t=0}^{n} \gamma^t \hat{\mathbb{E}}_k[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_n]$$

Now, computation is $O(j \cdot k)^n$

Trajectory Optimization

$$\underset{a_0,\ldots,a_n \in A}{\arg\max}\ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0,\ldots,a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_n]$$

Trick 2: Only simulate j actions and k states

$$\underset{a_0,\ldots,a_n \sim A^j}{\arg\max}\ \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \ldots, a_n] = \underset{a_0,\ldots,a_n \sim A^j}{\arg\max} \sum_{t=0}^{n} \gamma^t \hat{\mathbb{E}}_k[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \ldots, a_n]$$

Now, computation is $O(j \cdot k)^n$

Question: Drawbacks?

Trajectory Optimization

$$\underset{a_0, \dots, a_n \in A}{\arg\max} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \dots, a_n] = \underset{a_0, \dots, a_n \in A}{\arg\max} \sum_{t=0}^{n} \gamma^t \mathbb{E}[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \dots, a_n]$$

Trick 2: Only simulate j actions and k states

$$\underset{a_0, \dots, a_n \sim A^j}{\arg\max} \mathbb{E}[\mathcal{G}(\boldsymbol{\tau}_n) \mid s_0, a_0, \dots, a_n] = \underset{a_0, \dots, a_n \sim A^j}{\arg\max} \sum_{t=0}^{n} \gamma^t \hat{\mathbb{E}}_k[\mathcal{R}(s_{t+1}) \mid s_0, a_0, \dots, a_n]$$

Now, computation is $O(j \cdot k)^n$

Question: Drawbacks?

Answer: Optimal action may not be sampled, results in less-optimal trajectory

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Trajectory Optimization

Trajectory optimization/MPC is an "older" method

Less popular in the past, limited by compute

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

https://www.youtube.com/watch?v=bjlT-6KVQ7U

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

https://www.youtube.com/watch?v=bjlT-6KVQ7U

https://www.youtube.com/watch?v=Kf9WDqYKYQQ

Trajectory Optimization

Trajectory optimization/MPC is an "older" method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

https://www.youtube.com/watch?v=bjlT-6KVQ7U

https://www.youtube.com/watch?v=Kf9WDqYKYQQ

https://youtu.be/QsM9C1U0oi4?si=29BOjZ1Oo6At_iFk&t=111

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

https://www.youtube.com/watch?v=bjlT-6KVQ7U

https://www.youtube.com/watch?v=Kf9WDqYKYQQ

https://youtu.be/QsM9C1U0oi4?si=29BOjZ1Oo6At_iFk&t=111

We use trajectory optimization/MPC in both car racing AND chess/go bots!

Trajectory Optimization

Trajectory optimization/MPC is an “older” method

Less popular in the past, limited by compute

With modern GPUs, we are using MPC more and more

https://www.youtube.com/watch?v=bjlT-6KVQ7U

https://www.youtube.com/watch?v=Kf9WDqYKYQQ

https://youtu.be/QsM9C1U0oi4?si=29BOjZ1Oo6At_iFk&t=111

We use trajectory optimization/MPC in both car racing AND chess/go bots!

Trajectory Optimization

To summarize trajectory optimization/MPC:

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)
- Results in theoretically optimal policy

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)
- Results in theoretically optimal policy
- In practice, make approximations that sacrifice optimality for tractability

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)
- Results in theoretically optimal policy
- In practice, make approximations that sacrifice optimality for tractability
- Computationally expensive, but requires **no training data**

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)
- Results in theoretically optimal policy
- In practice, make approximations that sacrifice optimality for tractability
- Computationally expensive, but requires **no training data**

Next time, we will see what happens when we don't know Tr

Trajectory Optimization

To summarize trajectory optimization/MPC:

- Model-based method (we must know Tr and $\mathcal{R}$)
- Results in theoretically optimal policy
- In practice, make approximations that sacrifice optimality for tractability
- Computationally expensive, but requires **no training data**

Next time, we will see what happens when we don't know Tr

I plan to release assignment 1 next lecture