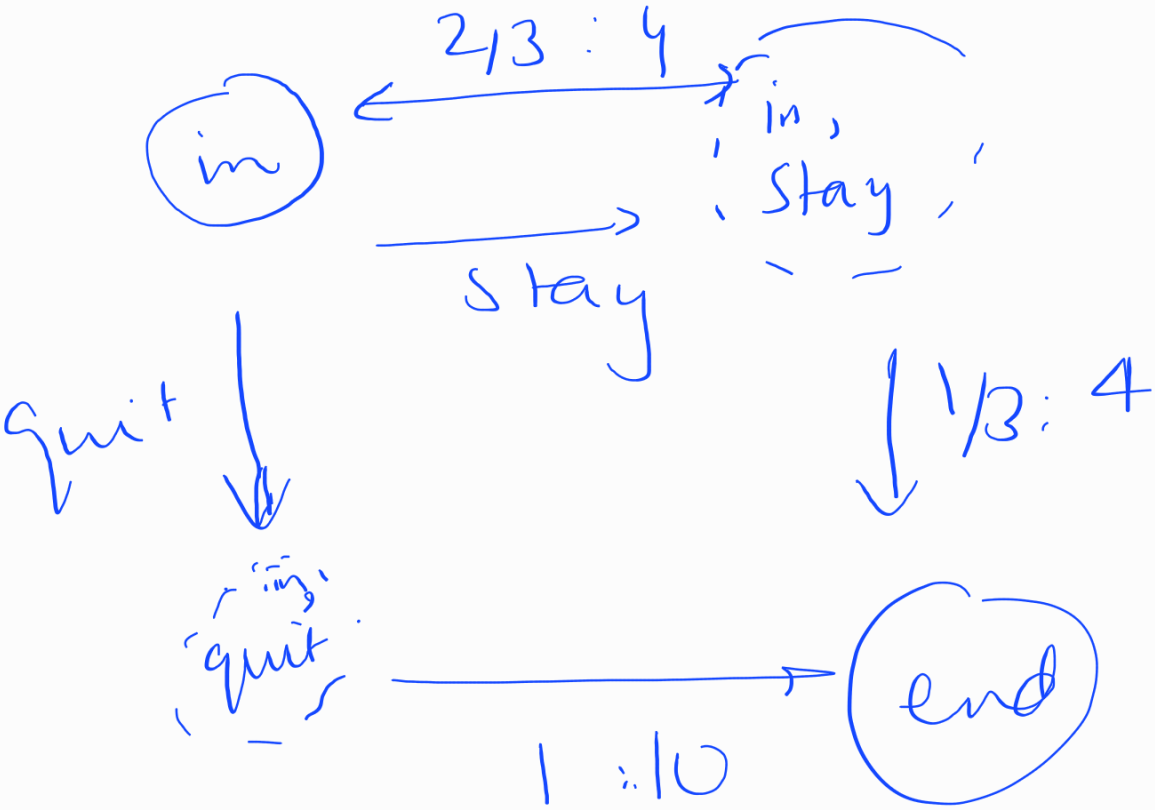
AI Theory, W7L1

Review: MDPs



$T(s, a, s') \rightarrow$ trans. prob.

$R(s, a, s') \rightarrow$ reward

What if we don't know the transition probabilities and rewards?

$$T(s, a, s') \longrightarrow \hat{T}(s, a, s')$$

$$R(s, a, s') \longrightarrow \hat{R}(s, a, s')$$

$\hat{T}$, $\hat{R}$ are estimated transition probabilities and rewards.

Definition of Episode:

At this point, it is helpful to

define "episode".

A step of policy in real world is called <u>episode</u>.

Let's define two episodes and try to learn transition probabilities.

In; stay; 4, In; stay; 4, end

In; stay; 4, end

If we take transition prob. and reward out of MDP, it is called <u>Reinforcement Learning</u>.

Prof Andrew Ng's autonomous helicopter is a landmark in RL.

Back to that example.

observing the environment, we see the following fact.

In, stay → In : 1 time

In, stay → End : 2 time

Let's run another episode.

→ In, stay, 4, In, stay, 4, In, stay, 4, End

$$\hat{T}(In, stay, In) = \frac{Count(In, stay, In)}{Count(In, stay)}$$

$$= \frac{3}{6}$$

If we add another episodes, counts also get updated.

Is it the best way to deal with estimated transition probabilities?

The law of probability is that a lot of probabilities converge uptil the real probability.

The reward can be calculated through average / weighted average.

This algorithm is called

Model Based Monte Carlo Simulation.

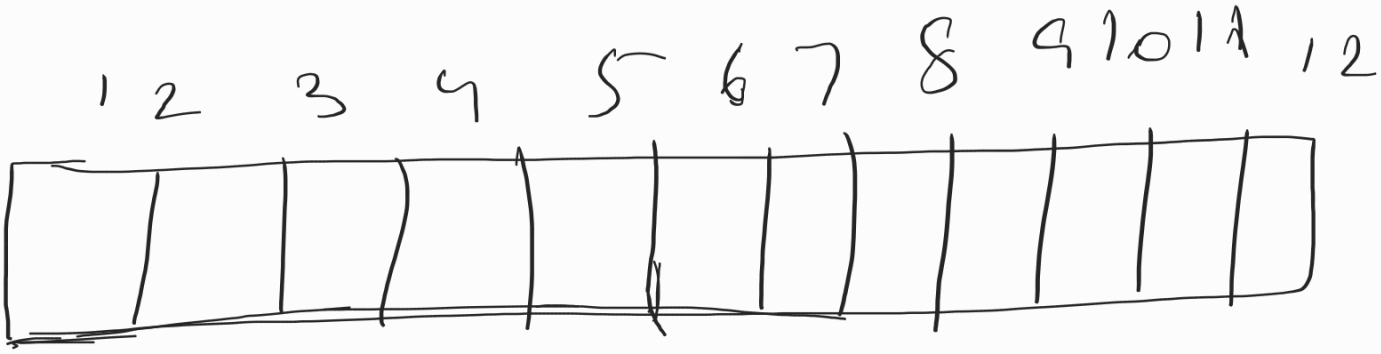
This algorithm has an issue.

This algorithm, due to low probability, will never explore "end" state, even though the reward can be greater at that point.

The solution is more exploration.

Although, in real world, exploitation is done.

Practical demonstration of estimating transition probability using Reinforcement learning:



A B → actions

in the form of

↳ (State, action, Reward)

State : 1

Reward : 0

<u>Episode 1</u> : (1, –, 0)

(5, A, –5), (8, A, 0), (3, B, 50)

<u>Episode 2</u> : (1, —, 0)

(12, A, 100)

Episode 3: (1, —, 0)

(11, B, −5), (4, B, 0), (4, B, 0)

Episode 4: (1, —, 0)

(5, B, −5), (7, B, −10), (11, B, −15),

(10, B, −10), (8, A, −5),

(6, B, −45)

Rewards Table

(1, A, 1) : N/A

(1, A, 2) : N/A

(1, A, 3) : N/A

(1, A, 4) : N/A

(1, A, 5) = −5

(1, B, 1) : N/A

(1, B, 2) : N/A

(1, B, 3) = −5

(1, B, 4) = N/A

(1, B, 5) = −5

-100

9, 12 yield negative rewards

↳ failure state

+50

3, 6 yield positive rewards

↳ failure state.

Hidden Rules were:

↳ odd state +5

even " −5

Nature

Estimator

Estimator:

estimates probabilities.

Nature:

Returns episodes.