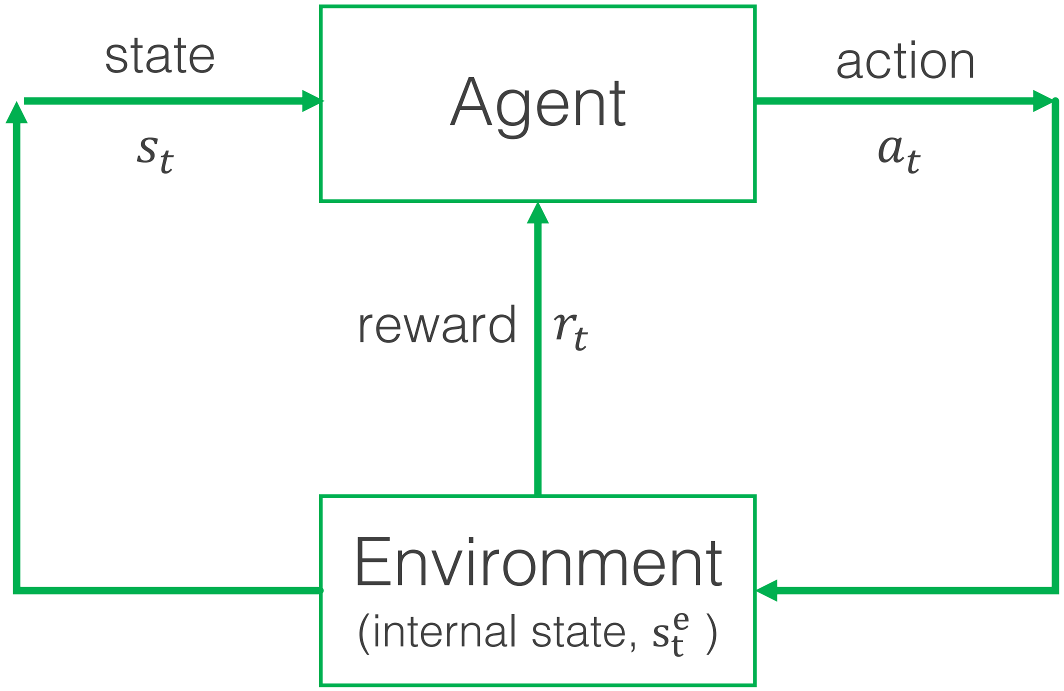
Reinforcement Learning II

Lecture 20

Reinforcement Learning Components



Policy (agent behavior), $\pi(s)$

- Determines action given current state
- Agent's way of behaving at a given time

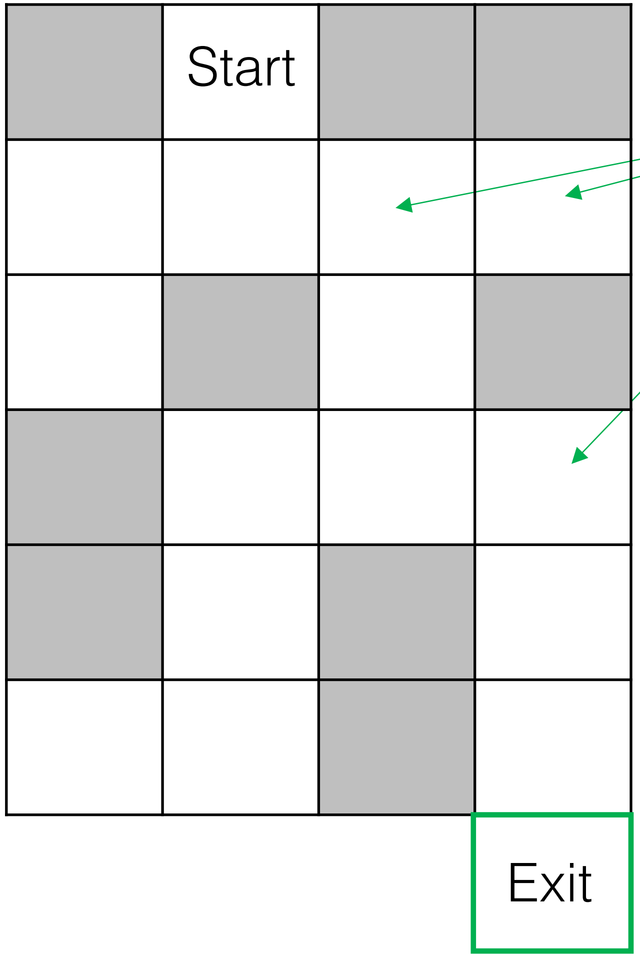
Reward function (the goal), r_t

- Objective is to **maximize total returns (cumulative reward)**

Value (expected returns), $v(s), q(s,a)$

- Expected returns from a state and following a specific policy

Maze Example: Policy, Value, and Reward



Each location in the maze represents a **state**

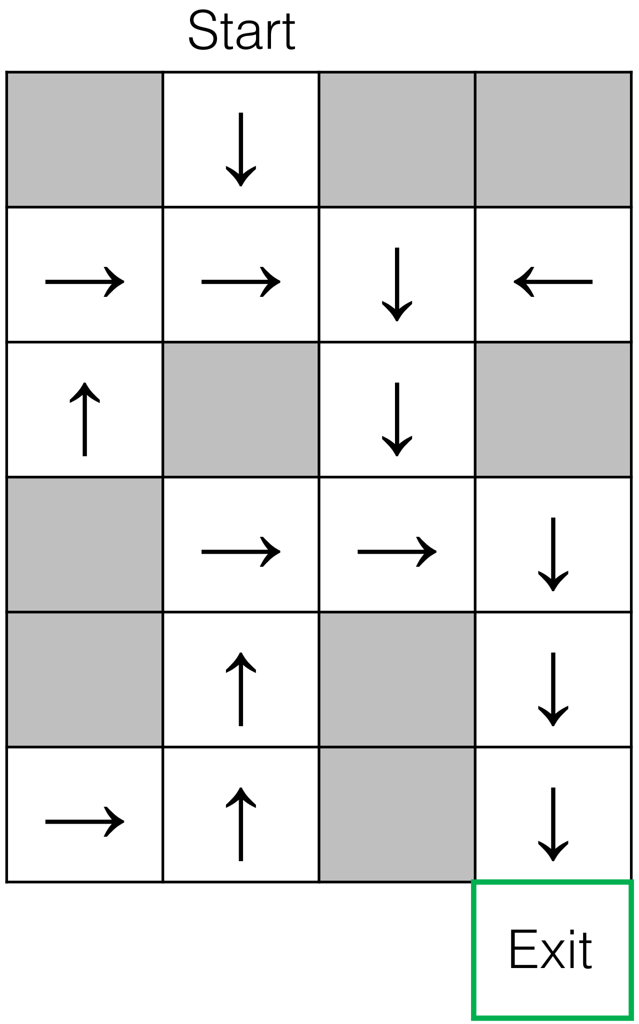
The **reward** is -1 for each step the agent is in the maze

Available **actions**: move ↑,↓,←,→ (as long as that path is not blocked)

Adapted from David Silver, 2015

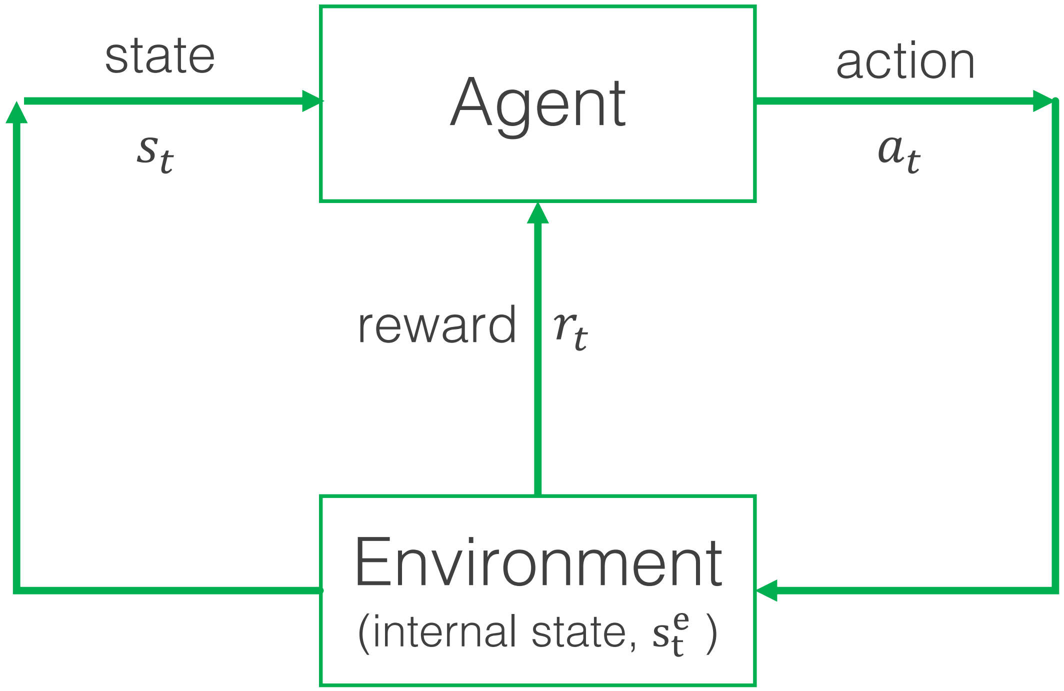
Policy $\pi(s)$

(which actions to take in each state)



Adapted from David Silver, 2015

Policy



Policy, $\pi(s)$

- Agent's way of behaving at a given time
- Maps state to actions

Deterministic: a $= \pi(s)$

Stochastic: $\pi(a|s) = P(a_t = a|s_t = s)$

Helps us "explore" the state space

RL tries to learn the "best" policy

Policy $\pi(s)$

(which actions to take in each state)

Reward r_t

(rewards are received as you transition OUT OF the state)

Start

	↓		
→	→	↓	←
↑		↓	
	→	→	↓
	↑		↓
→	↑		↓

Exit

Start

	-1		
-1	-1	-1	-1
-1		-1	
	-1	-1	-1
	-1		-1
-1	-1		-1

Exit

Adapted from David Silver, 2015

Goals and rewards

Rewards are the **only way** of communicating what to accomplish

Ex 1: Robot learning a maze

- 0 until it escapes, then +1 when it does
- -1 until it escapes (encourages it to escape quickly)

Ex 2: Robot collecting empty soda cans

- +1 for each empty soda can
- Negative rewards for bumping into things

Chess: what if we set +1 for capturing a piece?
(it may not win the game and still maximize rewards)

What you want achieved not **how**

Returns / cumulative reward

Episodic tasks (finite number, T, of steps, then reset)

$$G_t = r_{t+1} + r_{t+2} + \cdots + r_T$$

Continuing tasks with discounting ($T \rightarrow \infty$)

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

where $0 \leq \gamma \leq 1$ is the discount rate

This makes the agent care more about immediate rewards

Policy $\pi(s)$

(which actions to take in each state)

Start

	↓		
→	→	↓	←
↑		↓	
	→	→	↓
	↑		↓
→	↑		↓
			Exit

Reward r_t

(rewards are received as you transition OUT OF the state)

Start

	-1		
-1	-1	-1	-1
-1		-1	
	-1	-1	-1
	-1		-1
-1	-1		-1
			Exit

State Value $v_\pi(s)$

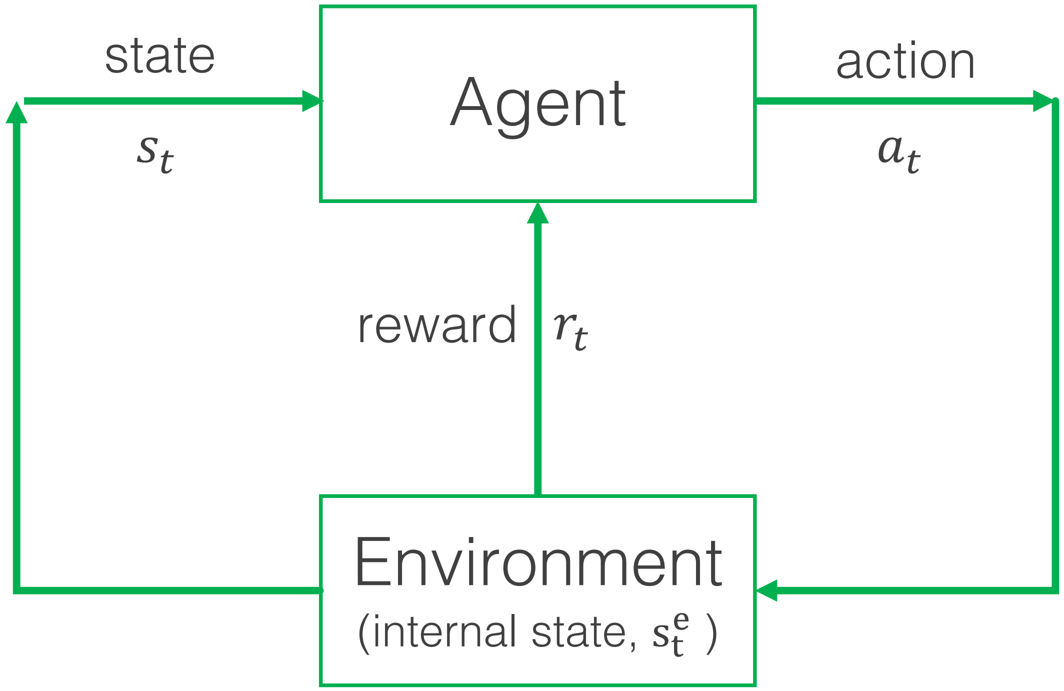
(expected cumulative rewards starting from current state **if** we follow the policy)

Start

	-8		
-8	-7	-6	-7
-9		-5	
	-5	-4	-3
	-6		-2
-8	-7		-1
			Exit

Adapted from David Silver, 2015

Value functions



State Value function, $v_\pi(s)$

- Expected returns (cumulative reward) assuming we follow policy π

$$v_\pi(s) = E_\pi[G_t | s_t = s]$$

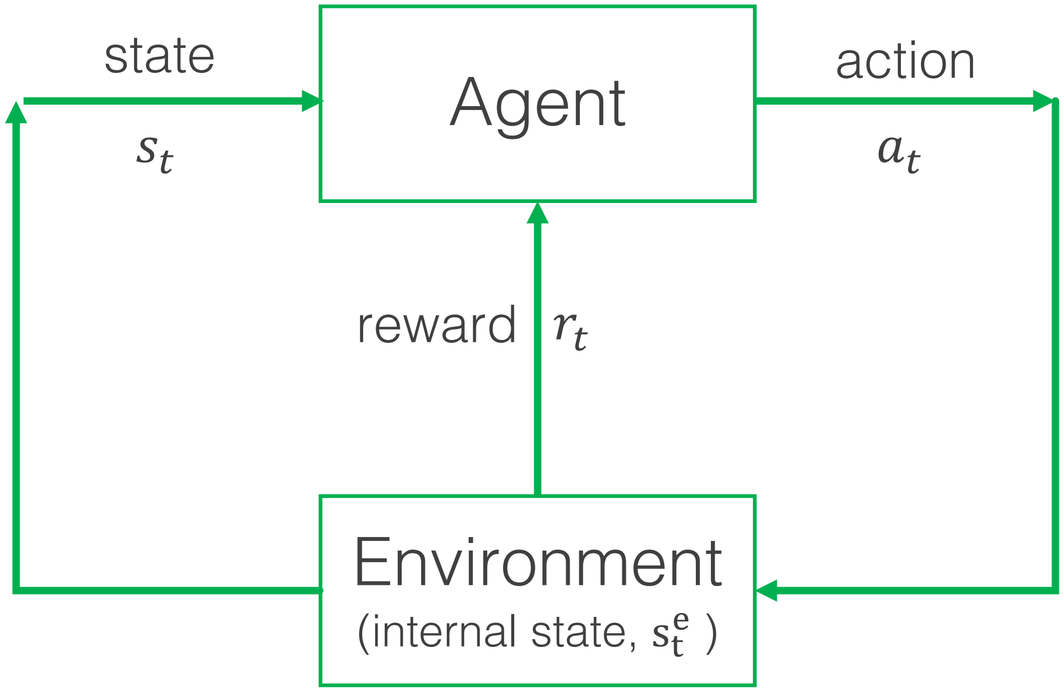
Action Value function, $q_\pi(s, a)$

- Expected returns from state, s, and taking action a, then follow policy π Total expected rewards

$$q_\pi(s, a) = E_\pi[G_t | s_t = s, a_t = a]$$

Where $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

Model



Model

Transition probabilities: predicts what state the environment will transition to next

$$P_{ss'}^a = P(s_{t+1} = s' | s_t = s, a_t = a)$$

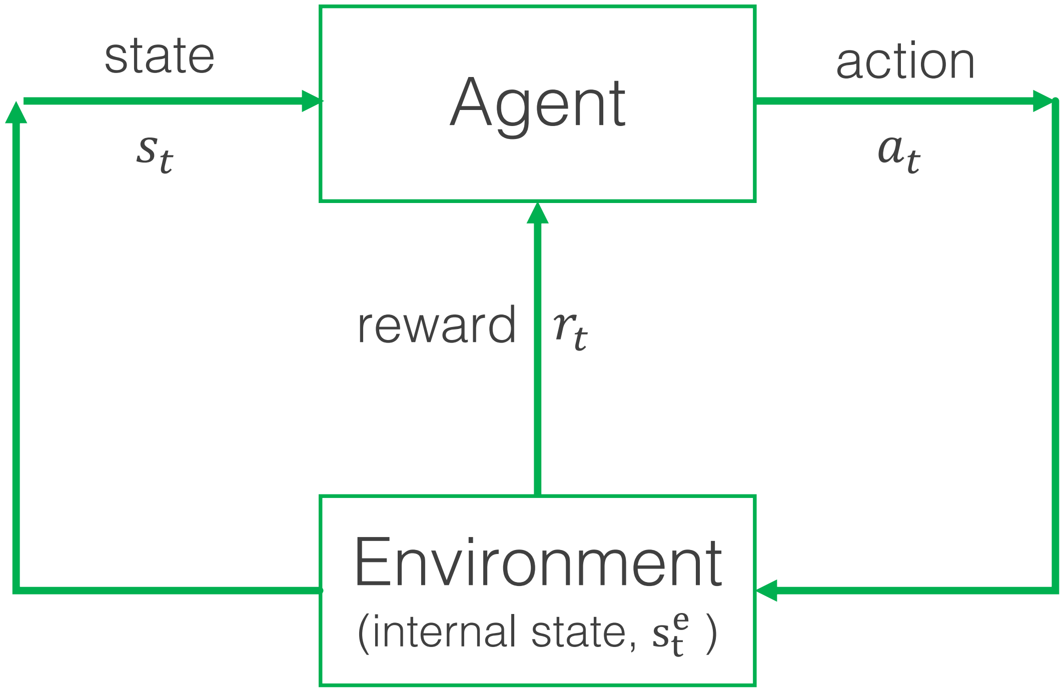
Expected Rewards: anticipates the next reward given an action

$$R_s^a = E[r_{t+1} | s_t = s, a_t = a]$$

"Planning" is the process of using these predictions

Model-based RL uses a model
Model-free RL does not use a model

Reinforcement Learning Components



Policy (agent behavior), $\pi(s)$

- Determines action given current state
- Agent's way of behaving at a given time

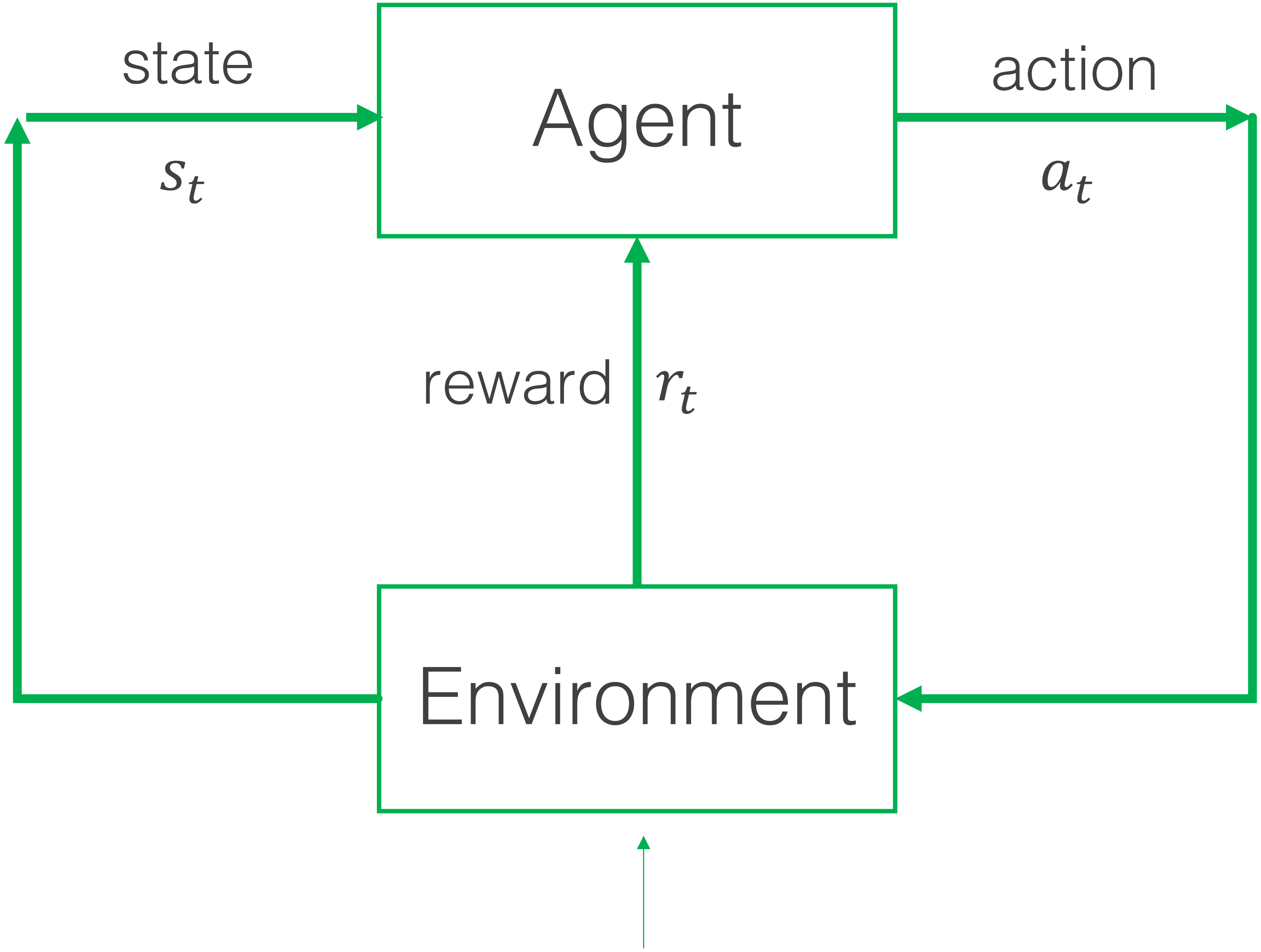
Reward function (the goal), r_t

- Objective is to **maximize total returns (cumulative reward)**

Value (expected returns), $v(s, a), q(s)$

- Expected returns from a state and following a specific policy

Environment



Markov Decision Process
(assumed form for most RL problems)

Goal

Maximize returns (expected rewards)

Find the best policy to guide our actions in an environment

Here, environment = Markov Decision Process

Reinforcement Learning

Learning strategy

Knowledge of **Environment**	Model-based (planning)	Model-free
No knowledge Must learn from experience		
Perfect knowledge Known MDP	**Dynamic Programming** Policy iteration Value iteration	

Reinforcement Learning

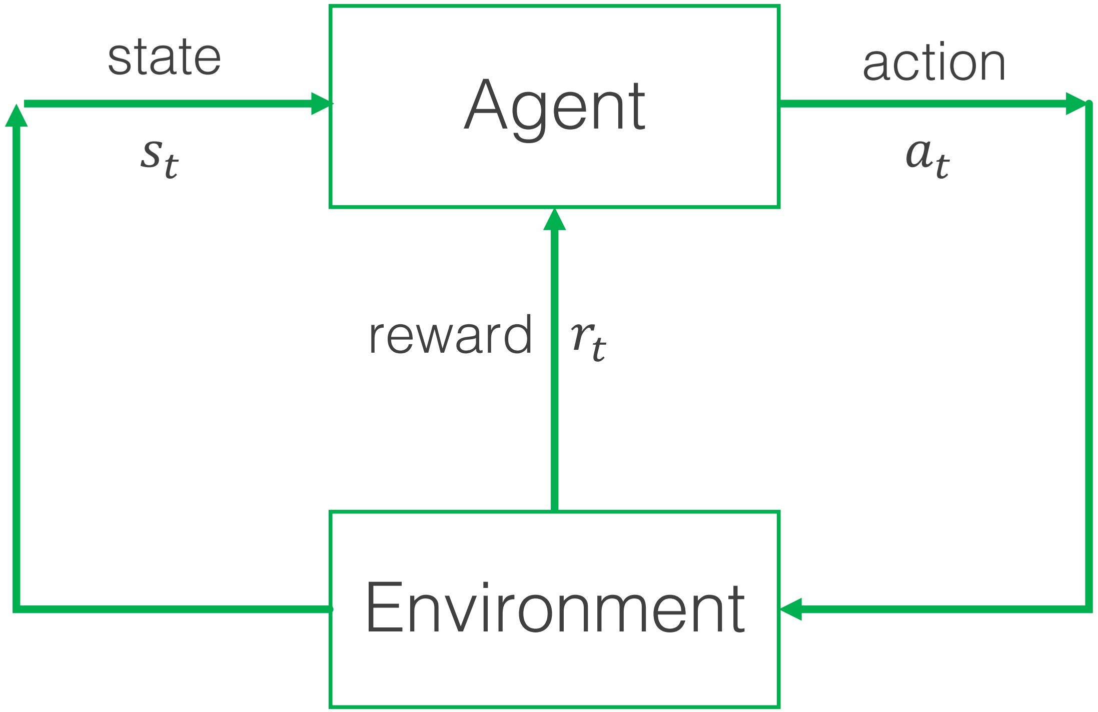
Policy evaluation
(prediction)

$$\pi \rightleftarrows v$$

Policy improvement
(optimization)

History

The record of all that has happened in this system



Step 0: s_0, a_0

Step 1: r_1, s_1, a_1

Step 2: r_2, s_2, a_2

$\vdots$

Step T: r_t, s_t, a_t

History at time t : $H_t = \{s_t, a_t, r_{t-1}, s_{t-1}, a_{t-1}, \ldots r_1, s_1, a_1, s_0, a_0\}$

Markov property

Instead of needing the full history:

$H_t = \{s_t, a_t, r_{t-1}, s_{t-1}, a_{t-1}, \dots r_1, s_1, a_1, s_0, a_0\}$

We can summarize everything in the current state

$H_t = \{s_t, a_t\}$

The future is independent of the past given the present

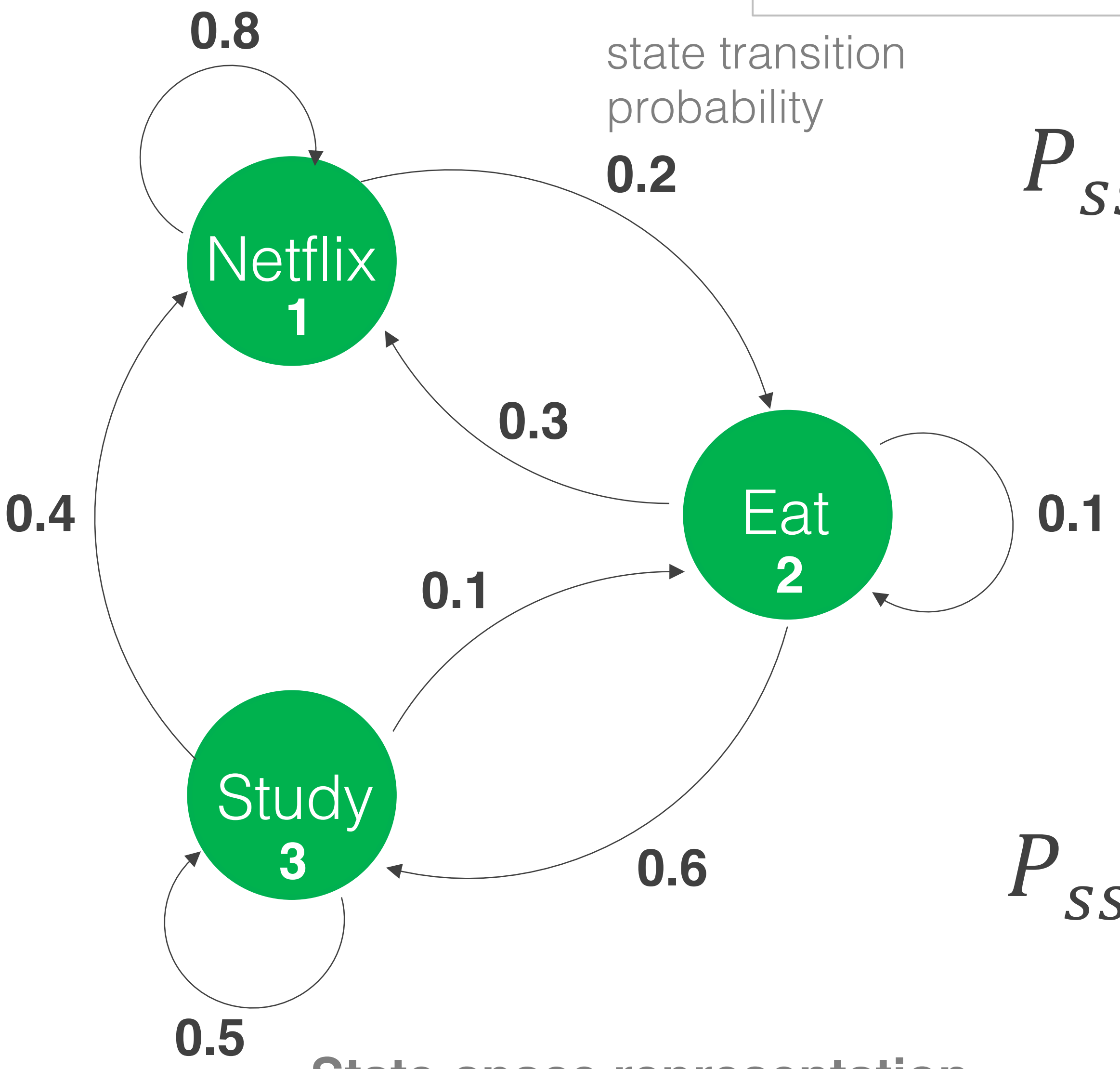
Another way of saying this is:

$P(s_{t+1}|s_t) = P(s_{t+1}|s_t, s_{t-1}, \dots, s_1, s_0)$

Markov Chains

Example: student life

Two components: $\{S, P\}$
State space, S
Transition matrix, P



State-space representation

State transition probabilities

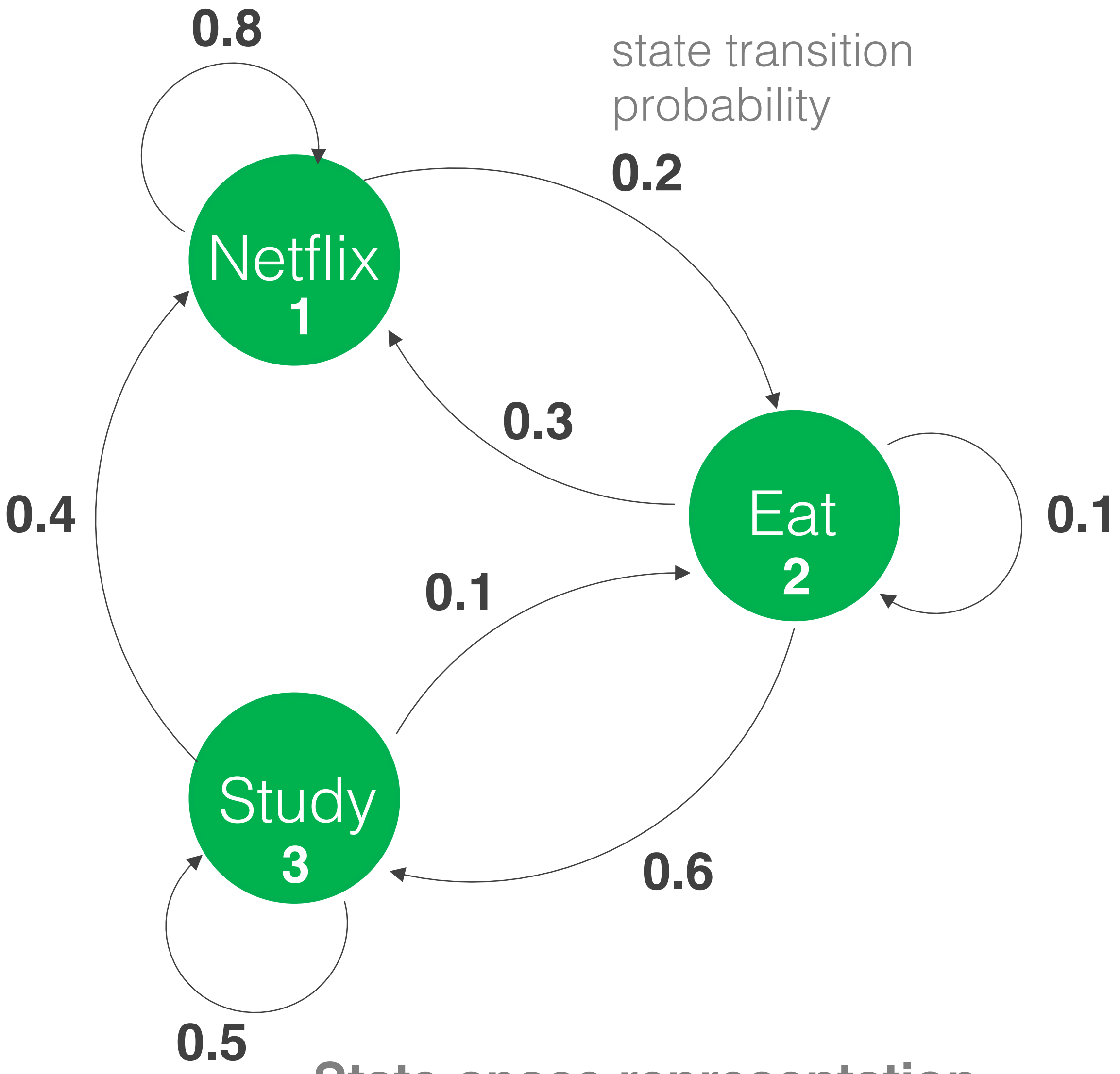
$$
P_{ss'} = \begin{array}{cc} & \text{To state} \\ & \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \text{From state} \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \end{array}
$$

Transitions out of each state sum to 1

$$
P_{ss'} = \begin{array}{cc} & \text{To state} \\ & \begin{array}{ccc} \text{Netflix} & \text{Eat} & \text{Study} \end{array} \\ \text{From state} \begin{array}{c} \text{Netflix} \\ \text{Eat} \\ \text{Study} \end{array} & \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.3 & 0.1 & 0.6 \\ 0.4 & 0.1 & 0.5 \end{bmatrix} \end{array}
$$

Markov Chains

Example: student life



State-space representation

If we start in state 1, what's the probability we'll be in each state after one step?

$$\begin{array}{cccc} & 1 & 2 & 3 \\ P_1 = & [0.8 & 0.2 & 0] \end{array}$$

This is the first row of the state transition probability matrix

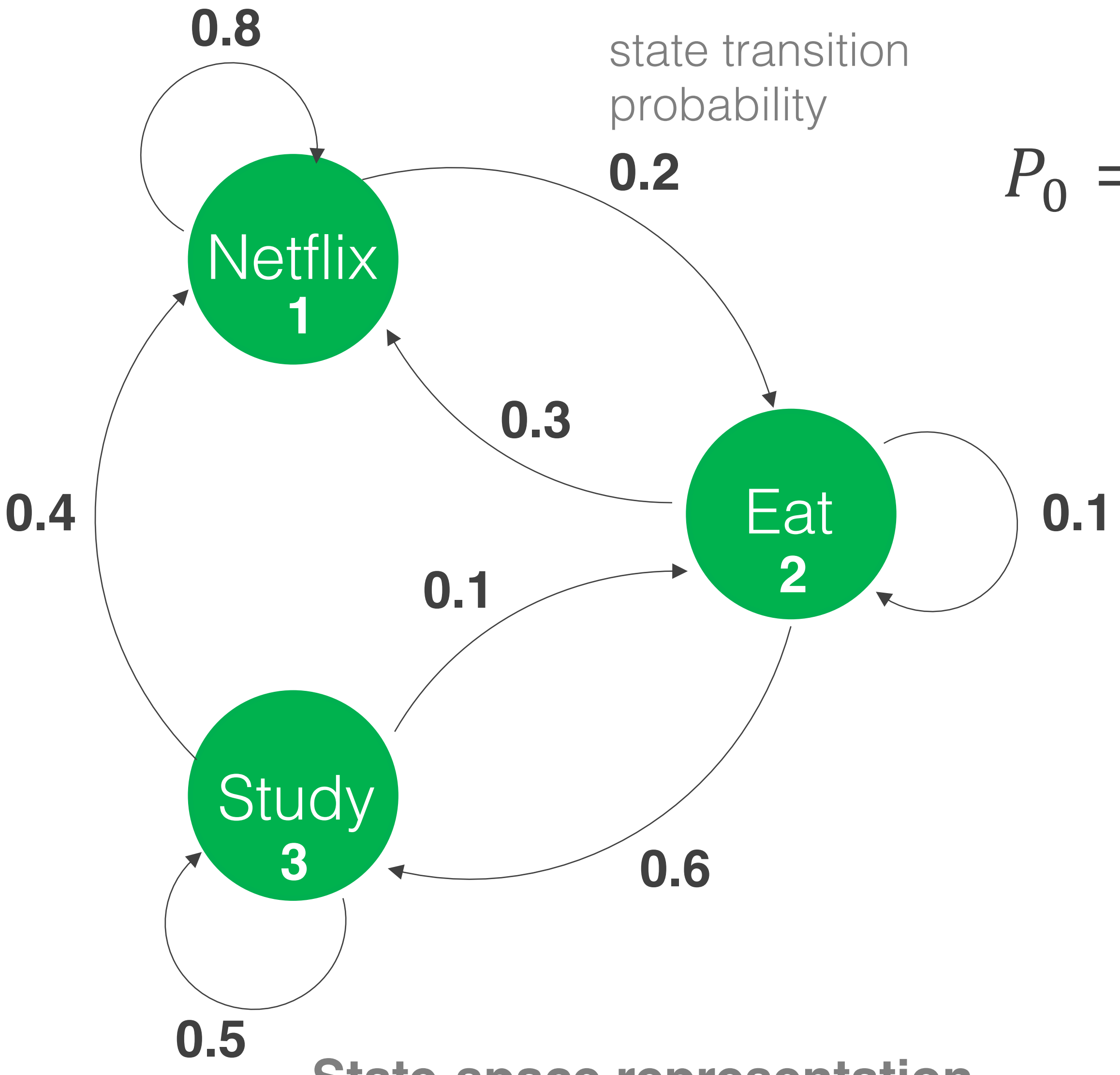
To state

$$P_{ss'} = \begin{array}{c|c} & \begin{array}{ccc} \text{Netflix} & \text{Eat} & \text{Study} \end{array} \\ \begin{array}{c} \text{Netflix} \\ \text{Eat} \\ \text{Study} \end{array} & \begin{bmatrix} \boxed{0.8 \quad 0.2 \quad 0} \\ 0.3 \quad 0.1 \quad 0.6 \\ 0.4 \quad 0.1 \quad 0.5 \end{bmatrix} \end{array}$$

From state

Markov Chains

Example: student life



State-space representation

If we started in state 1, we can calculate the probabilities of being in each state at step 1 as:

$$P_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^T \qquad P_1 = P_0 P_{ss'}$$

$$P_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.3 & 0.1 & 0.6 \\ 0.4 & 0.1 & 0.5 \end{bmatrix}$$

$$P_1 = \begin{bmatrix} 0.8 & 0.2 & 0 \end{bmatrix}$$

To state

From state

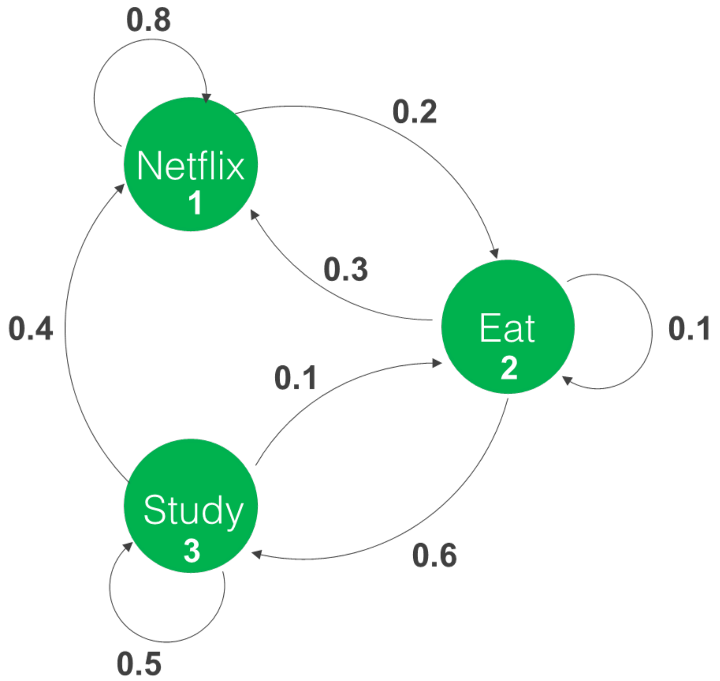
$$P_{ss'} = \begin{array}{c} \\ \text{Netflix} \\ \text{Eat} \\ \text{Study} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Netflix} & \text{Eat} & \text{Study} \end{array} \\ \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.3 & 0.1 & 0.6 \\ 0.4 & 0.1 & 0.5 \end{bmatrix} \end{array}$$

1 $P_1 = P_0 P_{ss'}$

$$P_1 = \begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}0.8 & 0.2 & 0\\0.3 & 0.1 & 0.6\\0.4 & 0.1 & 0.5\end{bmatrix}$$

$$P_1 = \begin{bmatrix}0.8 & 0.2 & 0\end{bmatrix}$$

$$P_0 = \begin{bmatrix}1\\0\\0\end{bmatrix}^T$$



2 $P_2 = P_1 P_{ss'} = P_0 P_{ss'} P_{ss'} = P_0 P_{ss'}^2$

$$P_2 = \begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}0.8 & 0.2 & 0\\0.3 & 0.1 & 0.6\\0.4 & 0.1 & 0.5\end{bmatrix}\begin{bmatrix}0.8 & 0.2 & 0\\0.3 & 0.1 & 0.6\\0.4 & 0.1 & 0.5\end{bmatrix}$$

$$P_2 = \begin{bmatrix}0.7 & 0.18 & 0.12\end{bmatrix}$$

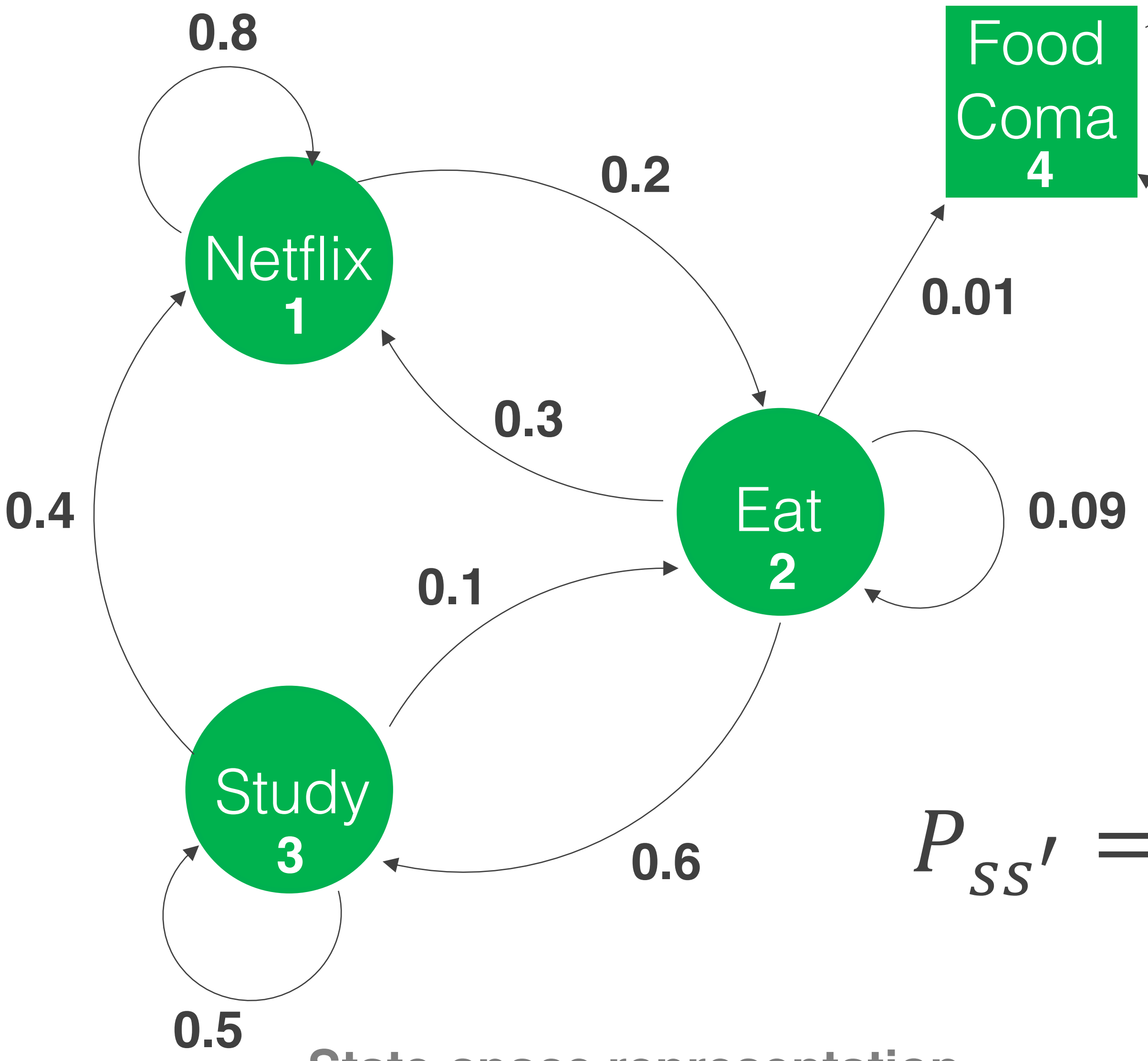
n $P_n = P_0 P_{ss'}^n$

As $n \to \infty$, we identify our steady state probabilities

$$P_\infty = \begin{bmatrix}0.64 & 0.16 & 0.20\end{bmatrix}$$

Markov Chains with absorbing state

Example: student life



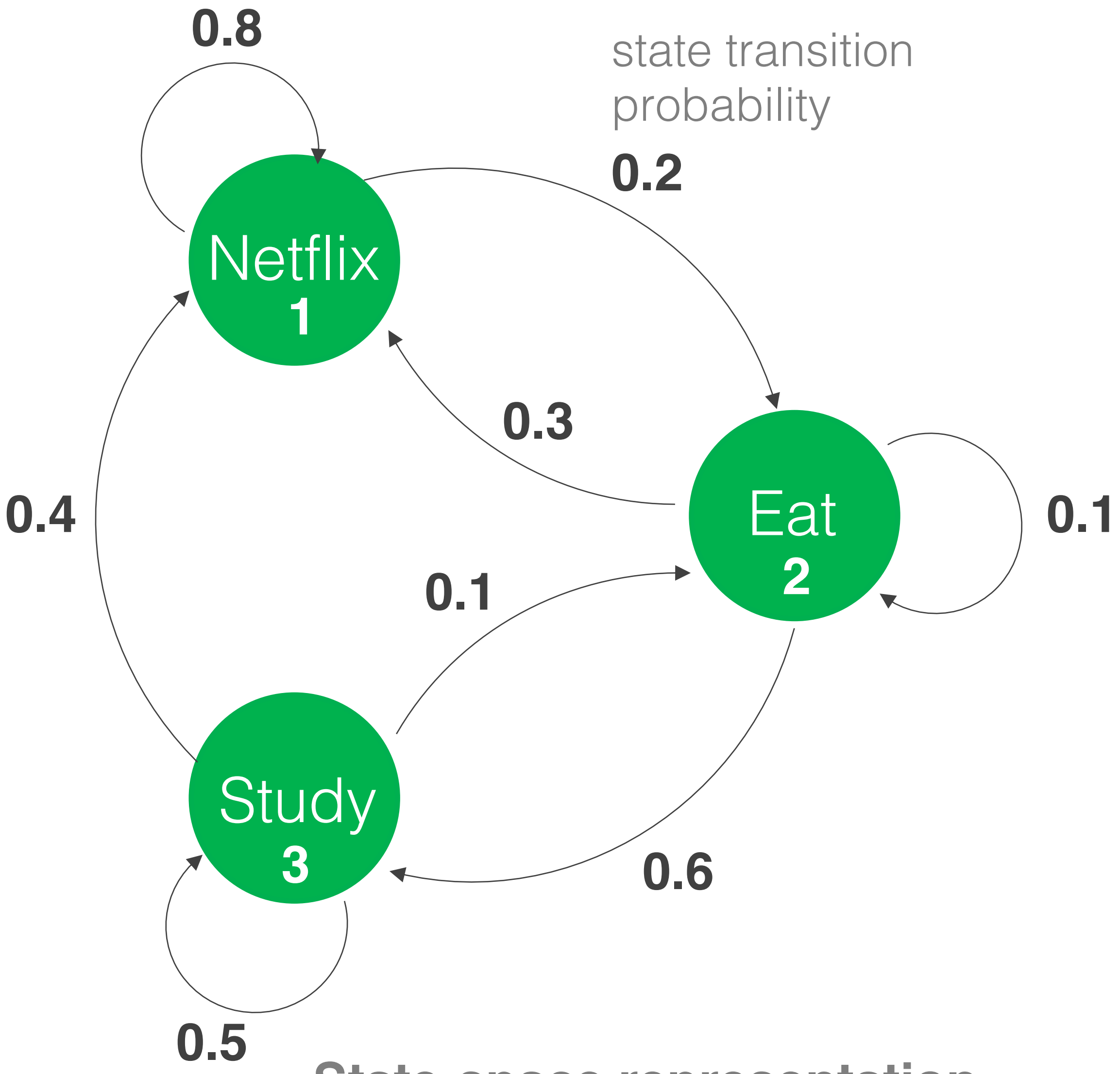
State-space representation

To state

$$P_{ss'} = \begin{array}{c c} & \begin{array}{cccc} \text{Netflix} & \text{Eat} & \text{Study} & \text{Food Coma} \end{array} \\ \text{From state} \begin{array}{c} \text{Netflix} \\ \text{Eat} \\ \text{Study} \\ \text{Food Coma} \end{array} & \begin{bmatrix} 0.8 & 0.2 & 0 & 0 \\ 0.3 & 0.09 & 0.6 & 0.01 \\ 0.4 & 0.1 & 0.5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Markov Chains

Example: student life



State-space representation

Markov chains can be used to represent sequential discrete-time data

Can estimate long-term state probabilities

Can simulate state sequences based on the model

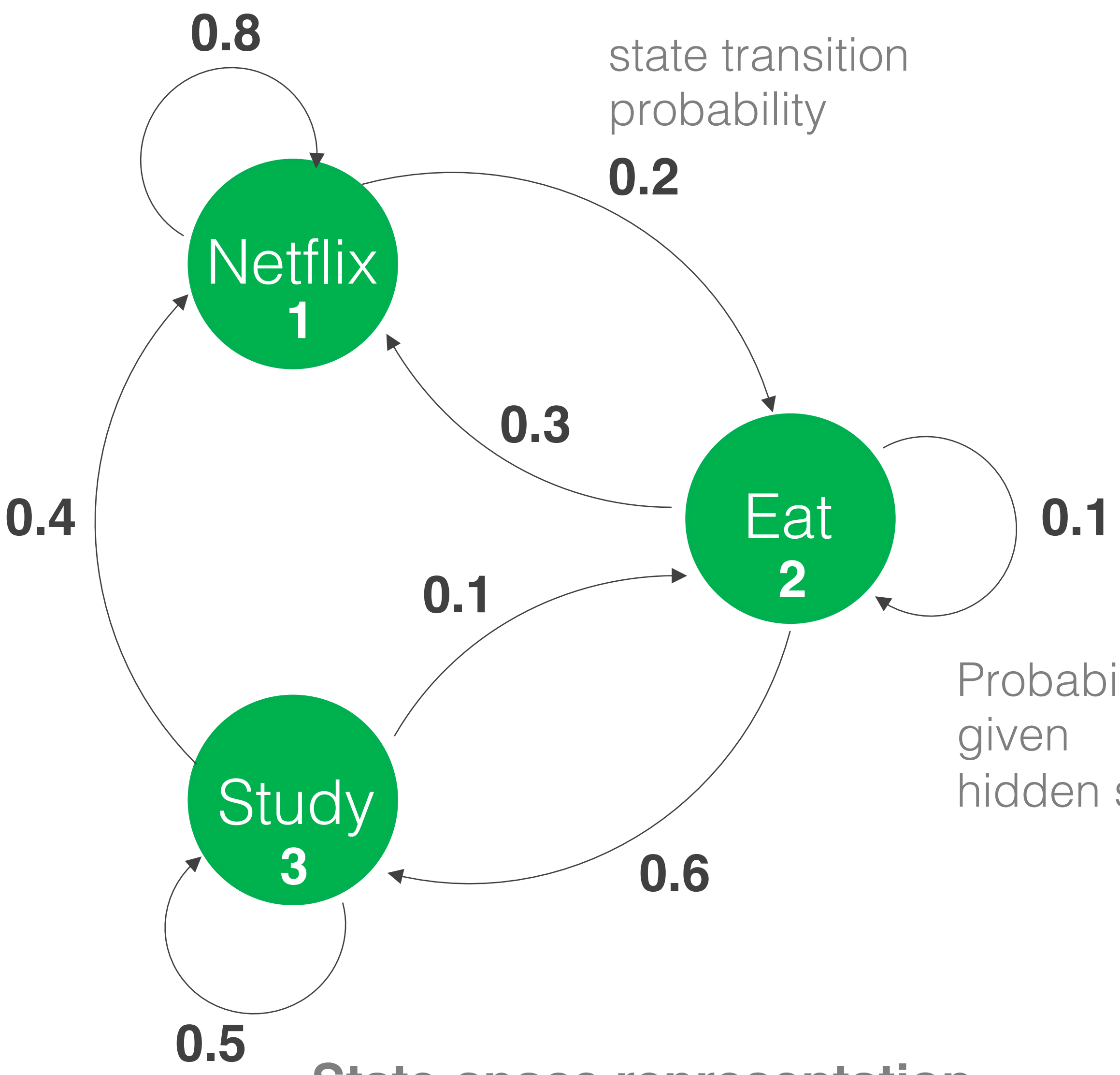
Markov property applies
(current state gives you all the information you need about future states)

$$P(s_{t+1}|s_t) = P(s_{t+1}|s_t, s_{t-1}, \dots, s_1, s_0)$$

Valid if the system is **autonomous** and the states are **fully observable**

Hidden Markov Models

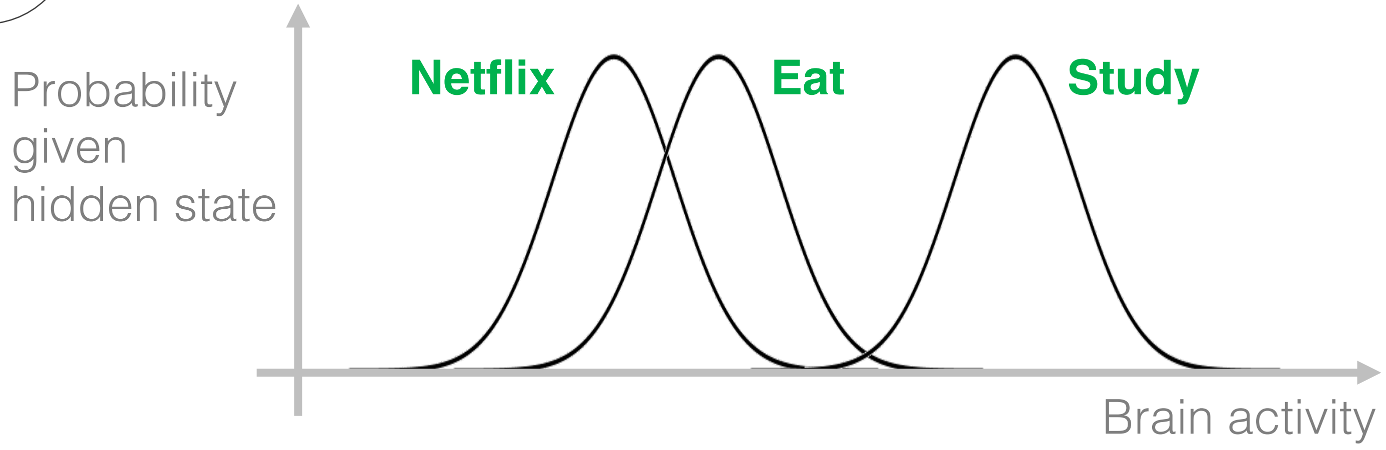
Example: student life



State-space representation

What if we **don't directly observe what state** the system is in, but instead observe a quantity that depends on the state?

Example: the student wears an EEG monitor, and we see readings of brain activity.



States are **hidden** or **latent variables**

Markov Models

	States are **Fully Observable**	States are **Partially Observable**
Autonomous (no actions; make predictions)	Markov Chain & Markov Reward Process	Hidden Markov Model (HMM)
Controlled (can take actions)	Markov Decision Process (MDP)	Partially Observable Markov Decision Process (POMDP)

Applications

HMMs: time series ML, e.g. speech + handwriting recognition, bioinformatics

MDPs: used extensively for reinforcement learning

Building blocks for the full RL problem

1	Markov Chain	{state space S, transition probabilities P}
2	Markov Reward Process (MRP)	$\{S, P,$ + rewards R, discount rate $\gamma\}$ adds rewards (and values)
3	Markov Decision Process (MDP)	$\{S, P, R, \gamma,$ + actions $A\}$ adds decisions (i.e. the ability to control)

MDPs form the basis for most reinforcement learning environments

Adapted from David Silver, 2015