Reinforcement Learning

Lecture 2

Markov Decision Processes

Policies

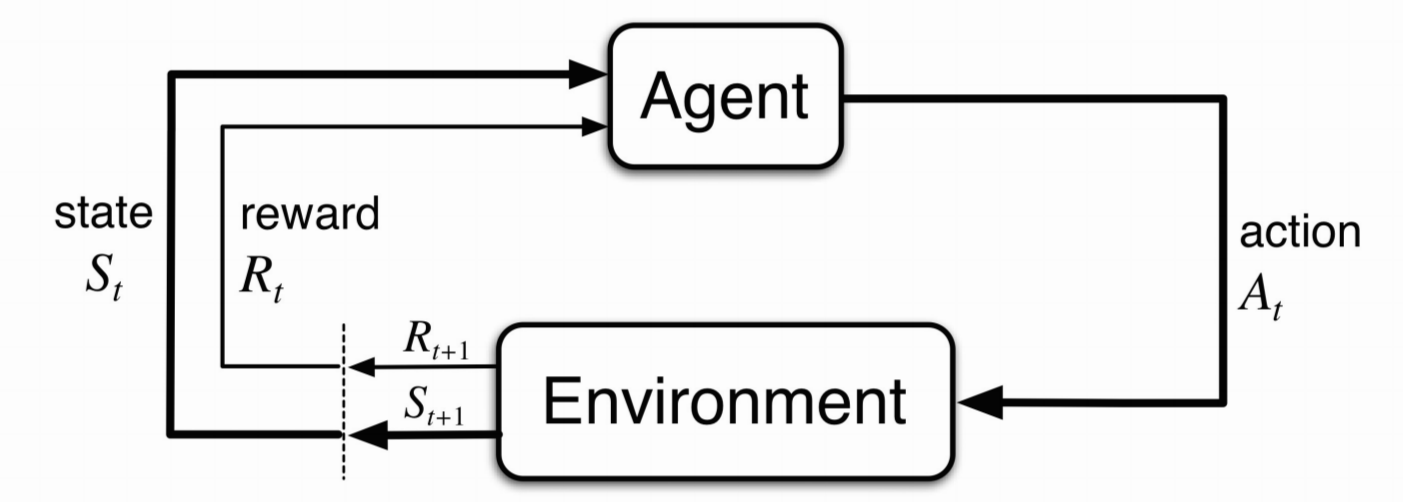
Return, Value Function, Bellman Equation

Robert Peharz

Institute of Theoretical Computer Science
Graz University of Technology
Winter Term 2023/24

Recap: Agent-Environment Interface

Image: Sutton & Barto



Last lecture: introduced on a high level

Today. Formal introduction of

Markov Decision Processes (MDPs),

the central notion in RL!

Recap: Chain Rule

Any joint $p(x_1, x_2, \ldots, x_D)$ can **always** be factorized as

$$p(x_1, x_2, \ldots, x_D) = p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2, x_1) \ldots p(x_D \mid x_1, \ldots, x_{D-1})$$

$$p(x_1, x_2, \ldots, x_D) = \prod_{i=1}^{D} p(x_i \mid x_1 \ldots x_{i-1})$$

(works with any ordering)

Recap: Conditional Independence

Random variables (vectors) $\underline{X}$ and $\underline{Y}$ are **conditionally independent** given $\underline{Z}$ if

- $p(\underline{x}, \underline{y} \mid \underline{z}) = p(\underline{x} \mid \underline{z})\, p(\underline{y} \mid \underline{z})$ $\quad \forall\, \underline{x}, \underline{y}, \underline{z}$
- $p(\underline{x} \mid \underline{y}, \underline{z}) = p(\underline{x} \mid \underline{z})$ $\quad \forall\, \underline{x}, \underline{y}, \underline{z}$
- $p(\underline{y} \mid \underline{x}, \underline{z}) = p(\underline{y} \mid \underline{z})$ $\quad \forall\, \underline{x}, \underline{y}, \underline{z}$

} equivalent conditions

Markov Property

Let $S_0, S_1, S_2, \ldots, S_t, \ldots$ be a **sequence** of random variables.

The sequence has the **Markov property**, if

$$p(S_{t+1} \mid S_t, S_{t-1}, \ldots, S_0) = p(S_{t+1} \mid S_t)$$

i.e., if S_{t+1} is **conditionally independent** of $\{S_{t-1}, S_{t-2}, \ldots, S_0\}$ given S_t.

- $p(S_0, S_1, S_2, S_3, \ldots) = p(S_0)\, p(S_1 \mid S_0)\, p(S_2 \mid S_1)\, p(S_3 \mid S_2) \ldots$
- "The future is independent of the past given the present"
- The current state S_t holds the same amount of information about the future as the whole history $\{S_0, S_1, S_2, \ldots, S_{t-1}, S_t\}$

Image: wikipedia.org

Markov Decision Process (MDP)

A **Markov Decision Process (MDP)** is a 3-tuple

$$(\mathcal{S}, \mathcal{A}, p)$$

where

$\mathcal{S}$ is a **state space** (**set of states** of environment)

$\mathcal{A}$ is an **action space** (**set of actions** of agent)

p is called the **dynamics** of the MDP. Formally, it is a **conditional probability distribution**

$$p(s', r \mid s, a)$$

where

$s' \in \mathcal{S}$ is the value of the **next state** S_{t+1},

$r \in \mathbb{R}$ is the value of the **reward** R_{t+1},

$s \in \mathcal{S}$ is the value of the **current state** S_t,

$a \in \mathcal{A}$ is the value of the **selected action** A_t.

- the **current state** S_t, the **next state** S_{t+1}, the **reward** R_{t+1}, and the **action** A_t are all treated as **random variables**, taking values $s \in \mathcal{S}$, $s' \in \mathcal{S}$, $r \in \mathbb{R}$, and $a \in \mathcal{A}$, respectively.
- no distribution over the initial state S_0 is defined. (we might assume an arbitrary one, or assume a fixed s_0)
- the next state S_{t+1} and reward R_{t+1} **depend** on the current state S_t and action A_t via the

dynamics $p(s', r \mid s, a)$

next state S_{t+1}, reward R_{t+1}, current state S_t, action A_t

- independent of $S_{t-1}, S_{t-2}, \ldots$ given S_t (Markov)
- a non-Markov Process can be made Markov, by making the state space "large enough" (e.g., by storing the entire history)

- here, we assume that the reward is earned in the **next** time step: R_{t+1} (notation from Sutton & Barto). In some literature, R_t is used, i.e. the current time step. It actually doesn't matter and is merely a re-indexing.
- when $\mathcal{S}$ and $\mathcal{A}$ are **finite** $\rightarrow$ **finite MDPs**. We will focus on finite MDPs, but many results carry over to infinite MDPs.
- we will see later, how the **action** A_t is selected. (via a so-called **policy**)
- technicality: not every action makes sense in each state.

  One solution is to consider **state dependent action sets** $\mathcal{A}(s)$ (set of actions which can be played is state $s \in \mathcal{S}$)

State Transitions

The dynamics $p(s', r \mid s, a)$ describe how the next state S_{t+1} and reward R_{t+1} are jointly generated:

$$S_{t+1}, R_{t+1} \sim p(s', r \mid S_t = s, A_t = a)$$

$\sim$ means "is sampled from" or "follows distribution"

By marginalizing R_{t+1}, we get the

state transition

integral for continuous r

$$p(s' \mid S_t = s, A_t = a) = \sum_r p(s', r \mid S_t = s, A_t = a)$$

It describes how the state evolves, irrespective of the reward:

$$S_{t+1} \sim p(s' \mid S_t = s, A_t = a)$$

Expected Reward

Similarly, we can marginalize the next state,

$$p(r \mid S_t = s, A_t = a) = \sum_{s' \in \mathcal{S}} p(s', r \mid S_t = s, A_t = a)$$

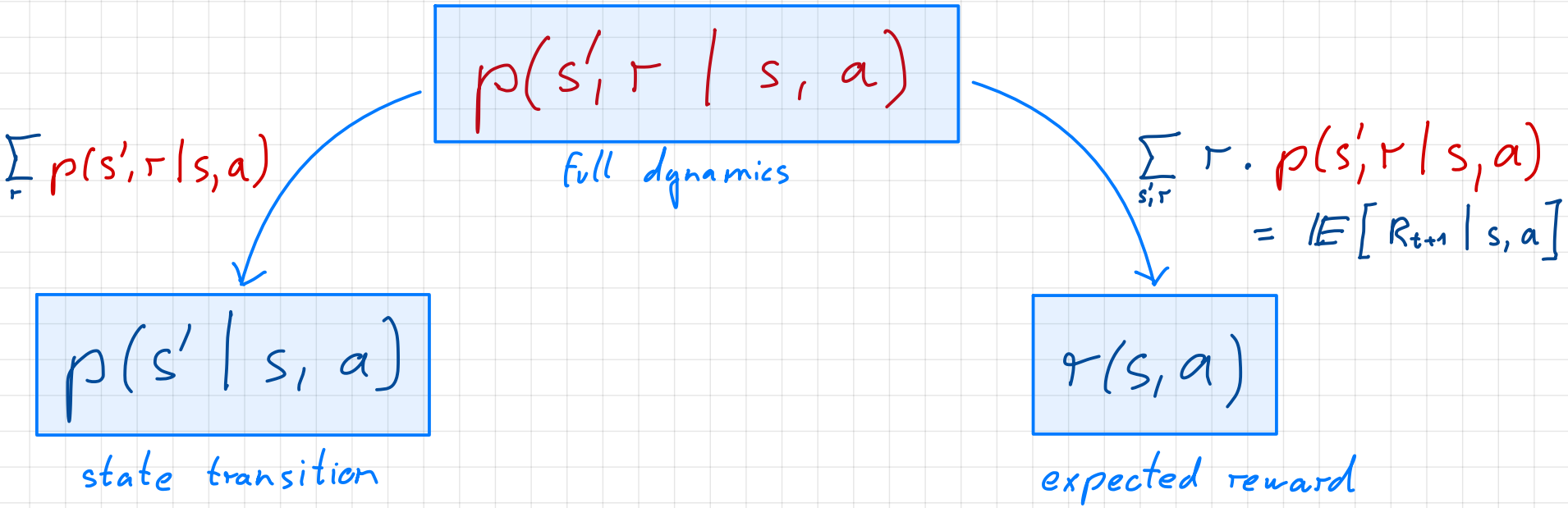
describing how the reward is generated, irrespective of the next state: $R_{t+1} \sim p(r \mid S_t = s, A_t = a)$

Usually, we are not interested in the full distribution $p(r \mid s, a)$, but only in its expectation:

$$r(s,a) := \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a] = \sum_r r \, p(r \mid S_t = s, A_t = a)$$

$r(s,a)$ is called the **expected reward function**.

Simplifying Dynamics



- $p(s', r \mid s, a)$ gives the general description of the MDP.
- $p(s' \mid s, a)$, $r(s, a)$ usually sufficient and easier to work with.
- thus, we also write $(\mathcal{S}, \mathcal{A}, p, r)$ for an MDP.

$\mathcal{S}$: state space
$\mathcal{A}$: action space
p: state transition
r: expected reward function

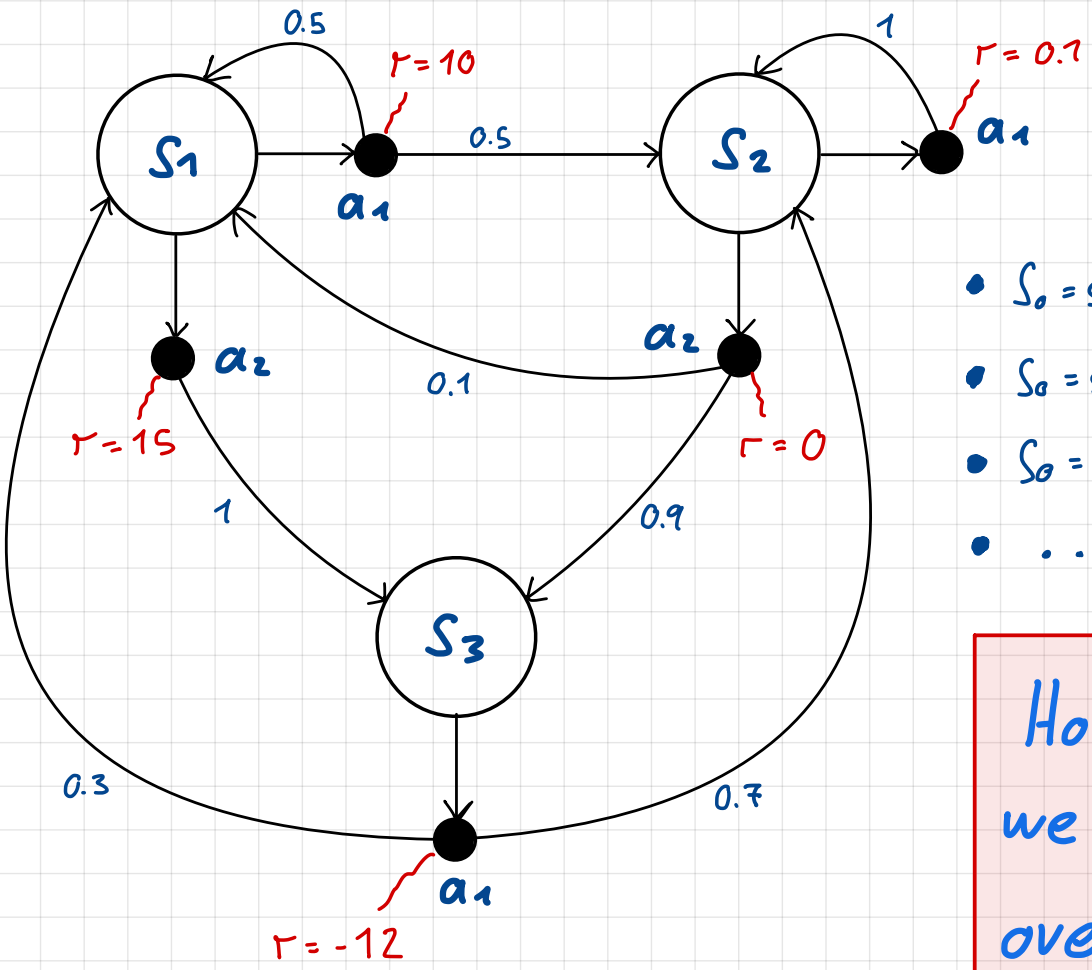
State Transition Diagrams

Visualization of (finite) MDPs

- each state value $s \in \mathcal{S}$ is an empty circle
- action values are small filled circles
- from each state value s draw arrows to $\mathcal{A}(s)$
- from each action draw arrows to possible next states $p(s' \mid s, a) > 0$
- label edges with transition probabilities
- label state-action pairs with (expected) reward

Example

$\mathcal{S} = \{s_1, s_2, s_3\}$, $\mathcal{A} = \{a_1, a_2\}$



Possible sequences:

- $S_0 = s_1, A_0 = a_1, R_1 = 10, S_1 = s_2, A_1 = a_2, R_2 = 0, S_2 = s_3, \ldots$
- $S_0 = s_2, A_0 = a_1, R_1 = 0.1, S_1 = s_2, A_1 = a_2, R_2 = 0, \ldots$
- $S_0 = s_1, A_0 = a_2, R_1 = 15, S_1 = s_3, A_1 = a_1, R_2 = -12, \ldots$
- ...

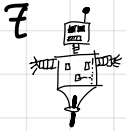
How to pick actions, so that we gather a lot of reward over time?

Note:

- $S_{t+1} = s_3$ follows deterministically from $S_t = s_1$ and $A_t = a_2$
- Similarly, $S_{t+1} = s_2$ when $S_t = s_2$ and $A_t = a_1$
- When $S_t = s_3$, only action is $A_t = a_1$ $(\mathcal{A}(s_3) = \{a_1\})$

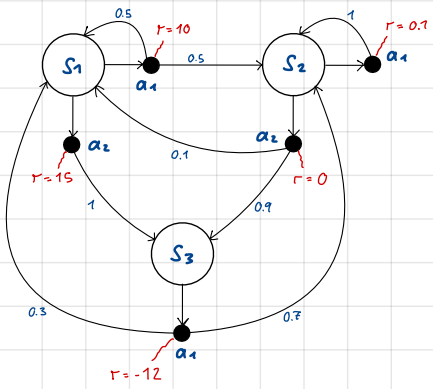
Example: Grid Worlds (didactic toy examples of MDPs)

- state $S_t = s \rightarrow$ agent in cell s
  $\mathcal{S} = \{1, 2, \dots, 16\}$
- actions $\mathcal{A} = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$ move agent to neighboring cell
- if it runs into border it stays in same cell
- e.g. $S_t = 7$, $A_t = \leftarrow$ yields $S_{t+1} = 6$
  $S_t = 3$, $A_t = \uparrow$ yields $S_{t+1} = 3$

1	2	3	4
5	6	7 	8
9	10	11	12
13	14	15	16

- in **terminal** states 1 & 16 agent stays forever (regardless of action)
- reward $R_{t+1} = -1$ after each step, except $R_{t+1} = 0$ for terminal states
- thus, agent should move to terminal states as quickly as possible

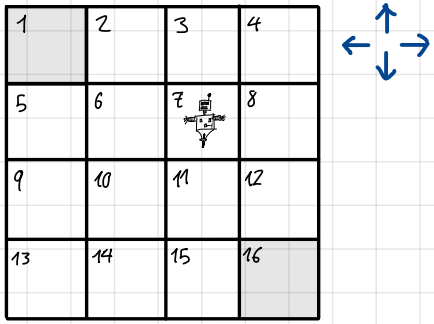
Episodic and Continuing Tasks



← a **continuing task**

goes on forever, produces infinite sequence

$S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$



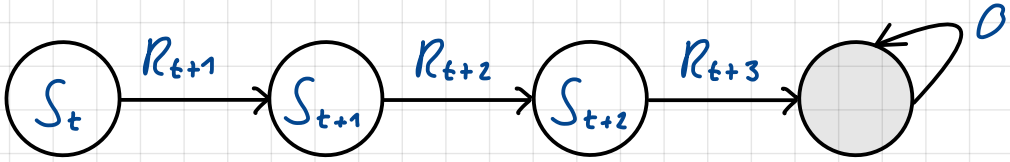
← an **episodic task**

terminates at states 1 and 16

generally, we observe many **episodes**

- 7, ←, -1, 6, ↑, -1, 2, ←, -1, 1 (stop)
- 10, ↓, -1, 14, ↓, -1, 14, ←, -1, 13, ..., 16 (stop)
- ...

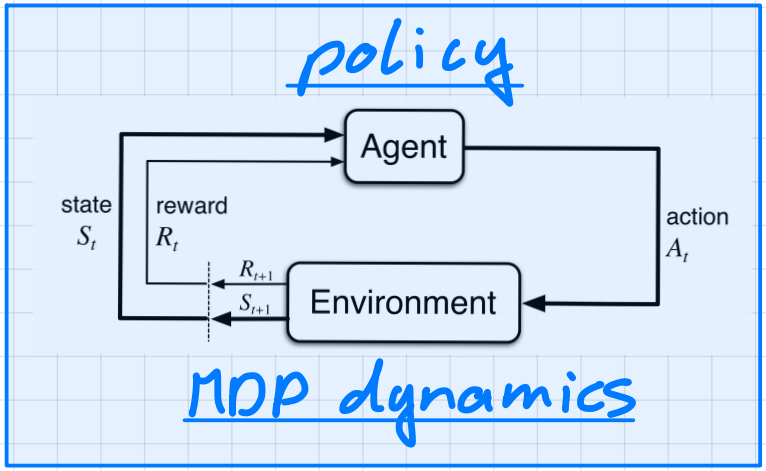
Technically, an episode can be seen as infinite sequence too:
Terminal states transit to themselves (for any action A_t) and yield $R_{t+1} = 0$.



Policies

- **MDP dynamics** describe the **environment**:
  - state dynamics, dependency on current state and action
  - reward function / distribution

- **Policies** describe the **agent**.
  - select an action, depending on current state S_t



A **deterministic policy** π is a **function**

$$\pi : \mathcal{S} \to \mathcal{A}$$

mapping states to actions.

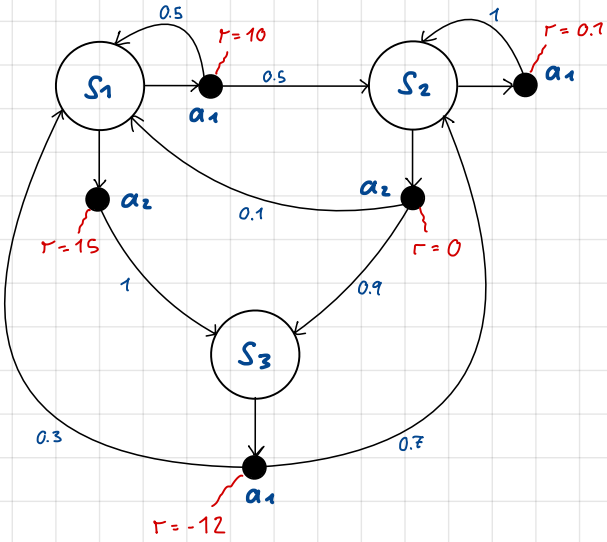
A **stochastic policy** π is a **conditional distribution**

$$\pi(a \mid s), \quad a \in \mathcal{A}, \quad s \in \mathcal{S}$$

over actions, conditional on any state.

Example: Policies

Let an MDP with state space $\mathcal{S} = \{s_1, s_2, s_3\}$ and action space $\mathcal{A} = \{a_1, a_2\}$ be given.



The function $\pi : \mathcal{S} \to \mathcal{A}$, defined as

$$\pi(s_1) = a_1, \quad \pi(s_2) = a_2, \quad \pi(s_3) = a_1$$

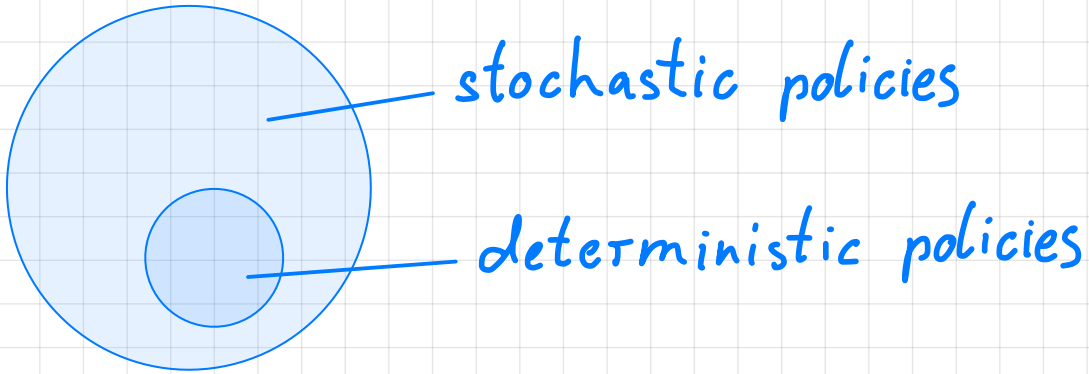
is a **deterministic policy**.

The conditional distribution $\pi(a|s)$, defined as

$$\pi(a_1|s_1) = 0.95 \quad \pi(a_1|s_2) = 0.25 \quad \pi(a_1|s_3) = 1$$
$$\pi(a_2|s_1) = 0.05 \quad \pi(a_2|s_2) = 0.75 \quad \pi(a_2|s_3) = 0$$

is a **stochastic policy**.

Deterministic and Stochastic Policies



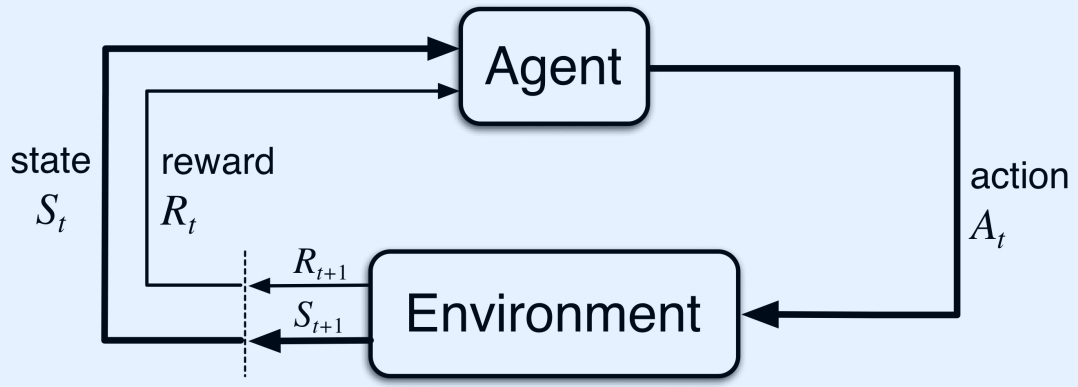
Any deterministic policy $\tilde{\pi}_{det}$ can be represented by a stochastic policy $\tilde{\pi}_{sto}$:

$$\tilde{\pi}_{sto}(a|s) := \begin{cases} 1 & \text{if } \tilde{\pi}_{det}(s) = a \\ 0 & \text{otherwise} \end{cases}$$

Thus, we will represent both deterministic and stochastic policies as conditional distributions $\tilde{\pi}(a|s)$.

The RL "Mechanics"

policy $\tilde{\pi}(a|s)$



MDP dynamics $p(s', r|s, a)$ $[p(s'|s,a), r(s,a)]$

- pick starting state S_0, $t = 0$
- repeat
  - sample $A_t \sim \tilde{\pi}(a|S_t)$ // policy
  - sample $S_{t+1}, R_{t+1} \sim p(s', r|S_t, A_t)$ // environment
  - $t \leftarrow t+1$
- this generates some random sequence

$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$

Return

What is a "good" policy?

What is the goal of RL?

The **return** at time t is defined as

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \dots = \sum_{k=0}^{\infty} R_{t+k+1}$$

The **discounted return** at time t is defined as

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

where $0 \leq \gamma \leq 1$.

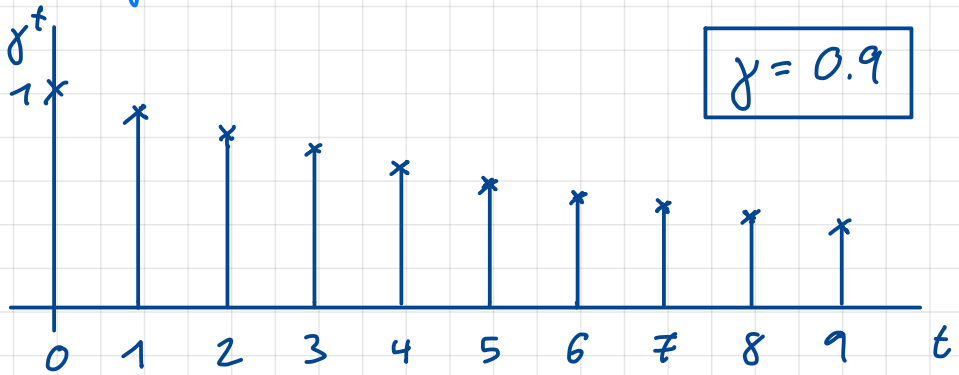
Note that G_t is a random variable.

Goal: **maximize expected (discounted) return** (w.r.t. $\tilde{\pi}$)

Discounting

Note that (undiscounted) return is a special case of discounted return with $\gamma = 1$.

For $\gamma < 1$, discounting introduces an exponentially decaying weight on the future rewards:



For $\gamma = 0$, $G_t = R_t$ which is the "myopic" (short-sighted) case: only the next reward is important, future rewards get ignored.

For $\gamma = 1$ (undiscounted case), we act maximally far-sighted: the reward in 1,000,000 time steps is as important as the next one.

Why Discounting?

Frankly, for mathematical convenience ...

Undiscounted return $G_t = \sum_{k=0}^{\infty} R_{t+k+1}$ might become ∞.

(How to optimize policy π, if any behavior could eventually lead to infinite return?)

Discounted return $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$ is bounded for $\gamma < 1$, if all rewards are bounded (they typically are).

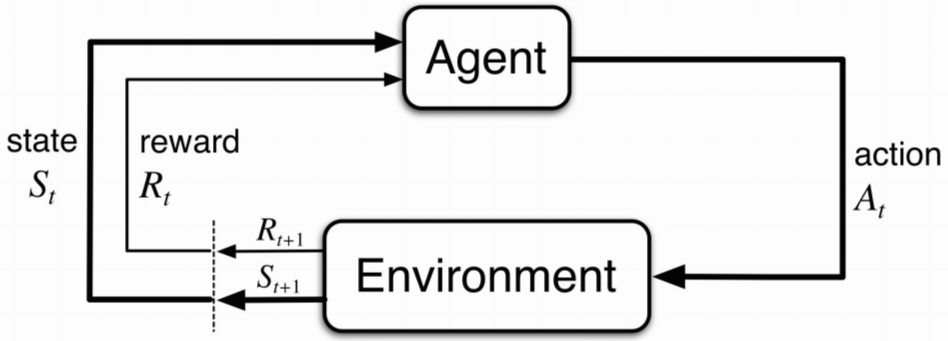
Further reasons:

- model misspecification
  MDP model not 100% accurate → focus on closer rewards
- financial thinking (?)
  A Euro today is more worth than a Euro tomorrow

Value Function v(s)

"How good is my policy?"

Given: MDP $(\mathcal{S}, \mathcal{A}, p, r)$ and policy $\pi(a|s)$.



- pick starting state S_0, $t = 0$
- repeat
  - sample $A_t \sim \pi(a|S_t)$
  - sample $S_{t+1}, R_{t+1} \sim p(s', r|S_t, A_t)$
  - $t \leftarrow t+1$

Running MDP with policy π leads to $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$

The **value function** $v_\pi : \mathcal{S} \to \mathbb{R}$ is defined as

$$v_\pi(s) := \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right]$$

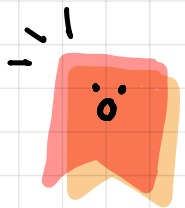
"The expected discounted return when following policy π, and starting from $s \in \mathcal{S}$ (t is any arbitrary time step)."

$\mathbb{E}_\pi$ is shorthand for $\mathbb{E}_{S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots}$

Example

- recall the **Grid World example**
- actions $\leftarrow, \rightarrow, \uparrow, \downarrow$ move agent
- reward -1 for every move, until a final state is reached
- in final state: reward 0 and stay in same state
- assume a random policy $\pi(a|s)$ which selects each action with 0.25 probability, regardless of s
- assume undiscounted return $G_t = \sum_{k=0}^{\infty} R_{t+k+1}$
- value function $v_\pi = \mathbb{E}_\pi[G_t | S_t = s]$ is shown here

0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0



How to compute this?

Recap: Iterated Expectations

$$\mathbb{E}[x] = \sum_x p(x)\, x$$

second Y involved

$$x, y \sim p(x, y)$$

$$\mathbb{E}[x] = \sum_x p(x)\, x$$

$$= \sum_x \sum_y p(x, y)\, x$$

marginal

$$= \sum_x \sum_y p(x \mid y)\, p(y)\, x$$

chain rule

$$= \sum_y p(y) \sum_x p(x \mid y)\, x = \mathbb{E}_y\left[\mathbb{E}_x[x \mid y]\right]$$

3 variables X, Y, Z $\quad x, y, z \sim p(X, Y, Z)$

$$\mathbb{E}[X] = \mathbb{E}_Z\left[\mathbb{E}_Y\left[\mathbb{E}_X[X \mid \cancel{Z,} Y] \mid Z\right]\right]$$

$$X \perp\!\!\!\perp Z \mid Y$$

Bellman Equation

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Recall definition of the value function

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right]$$

We can write v_π **recursively**:

$$\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right] \\
&= \mathbb{E}_\pi\left[\underbrace{\gamma^0}_{=1} R_{t+1} + \sum_{k=1}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \sum_{k=0}^{\infty} \gamma^{k+1} R_{t+k+2} \mid S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s\right]
\end{aligned}$$

Markov!

iterated expectation

$$= \mathbb{E}_{A_t, R_{t+1}, S_{t+1}}\left[R_{t+1} + \mathbb{E}_\pi[\gamma G_{t+1} \mid \cancel{S_t = s}, \cancel{A_t}, \cancel{R_{t+1}}, S_{t+1}] \mid S_t = s\right]$$

$$= \mathbb{E}_{A_t, R_{t+1}, S_{t+1}}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s\right]$$

Bellman Expectation Equation

$$v_\pi(s) = \mathbb{E}_{A_t, R_{t+1}, S_{t+1}} \left[R_{t+1} + \gamma \, v_\pi(S_{t+1}) \mid S_t = s \right]$$

Image: wikipedia.org

Concretely:

$$v_\pi(s) = \sum_a \pi(a|s) \left[\sum_{s',r} p(s', r | s, a) \left(r + \gamma \, v_\pi(s') \right) \right]$$

$r(s,a) = \mathbb{E}[r|s,a]$ expected reward function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a) \, r + \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a) \, \gamma \, v_\pi(s')$$

$p(s', a, r | s)$

$$= \underbrace{\sum_a \pi(a|s) \, r(s,a)}_{=: r(s)} + \sum_{s'} \underbrace{\sum_a \sum_r \pi(a|s) \, p(s', r | s, a)}_{=: p(s'|s)} \gamma \, v_\pi(s')$$

$r(s)$: expected reward from s

$p(s'|s)$: combined state transition

$$= r(s) + \gamma \sum_{s'} p(s'|s) \, v_\pi(s')$$