2- Reinforcement Learning Intro

Melih Kandemir

University of Southern Denmark
Department of Mathematics and Computer Science (IMADA)
kandemir@imada.sdu.dk

Fall 2022

The reward hypothesis

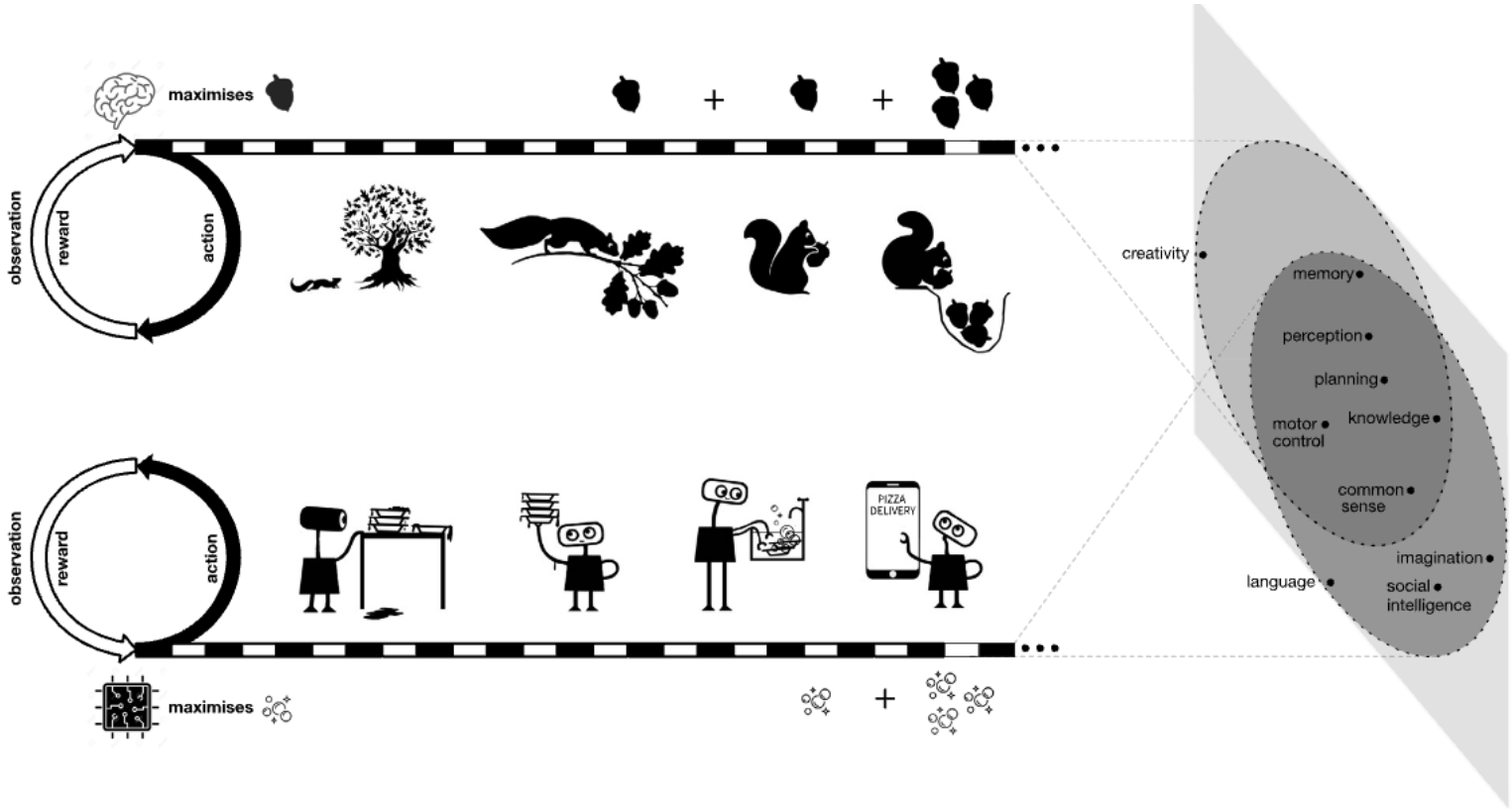
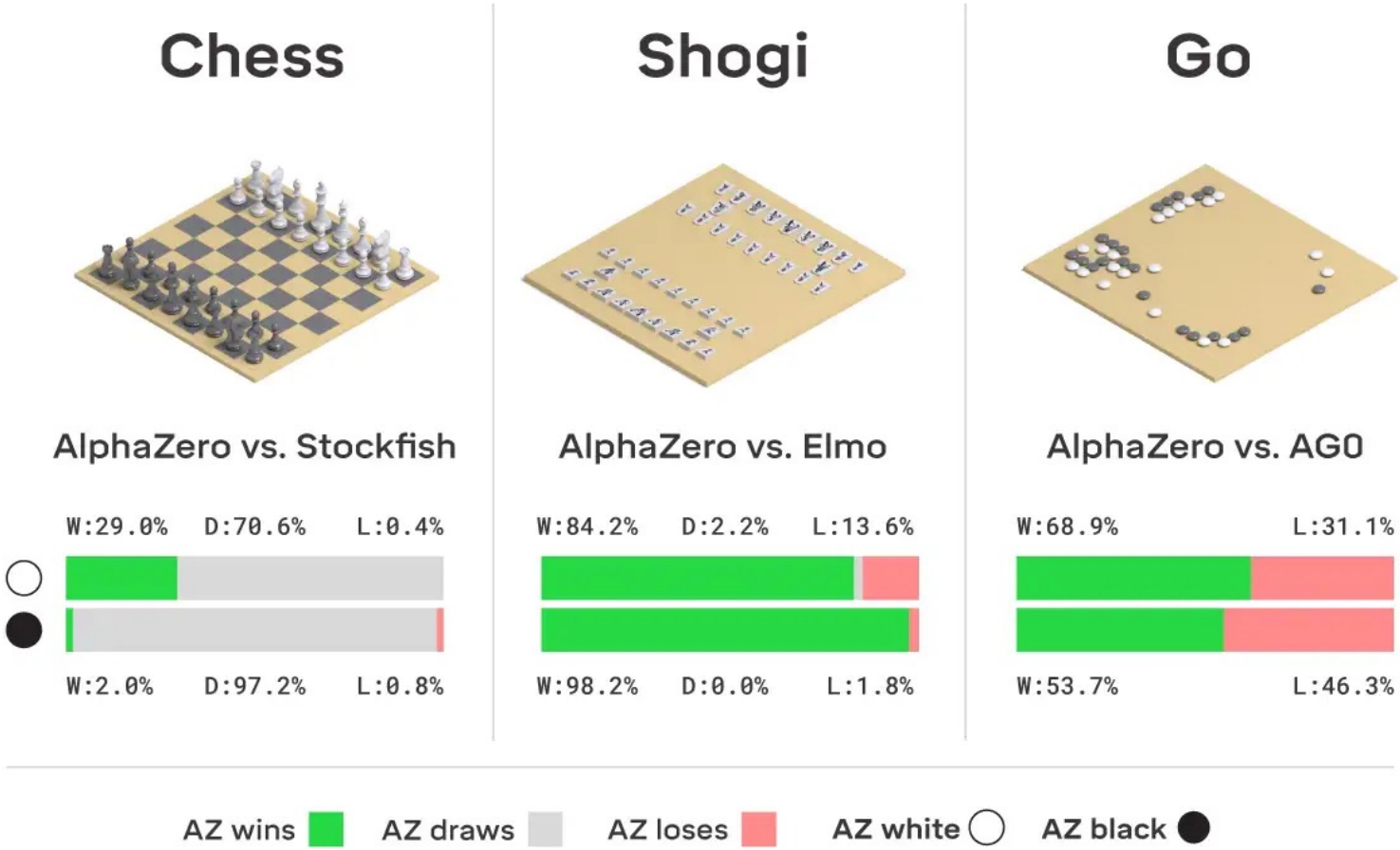


Fig. 1. The *reward-is-enough* hypothesis postulates that intelligence, and its associated abilities, can be understood as subserving the maximisation of reward by an agent acting in its environment. For example, a squirrel acts so as to maximise its consumption of food (top, reward depicted by acorn symbol), or a kitchen robot acts to maximise cleanliness (bottom, reward depicted by bubble symbol). To achieve these goals, complex behaviours are required that exhibit a wide variety of abilities associated with intelligence (depicted on the right as a projection from an agent's stream of experience onto a set of abilities expressed within that experience).

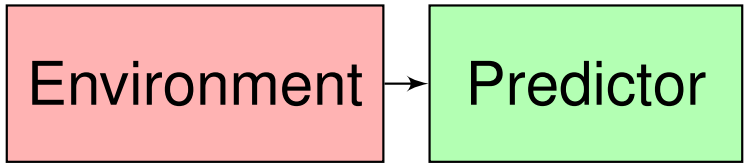
Figure: D. Silver et al., Reward is enough, Artif. Intl., 2021

and its evidence

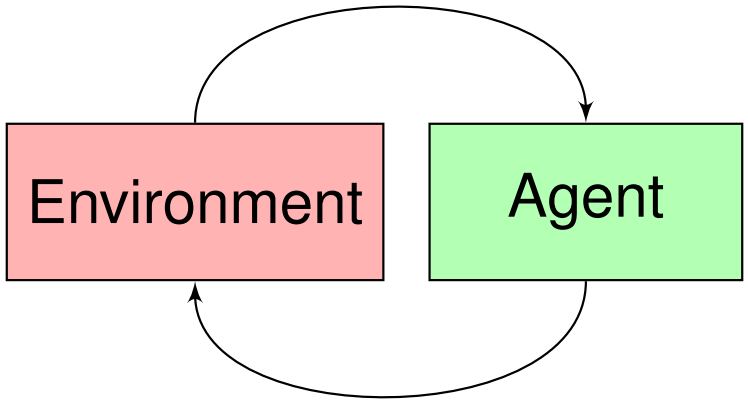


The reinforcement learning setup

Supervised learning



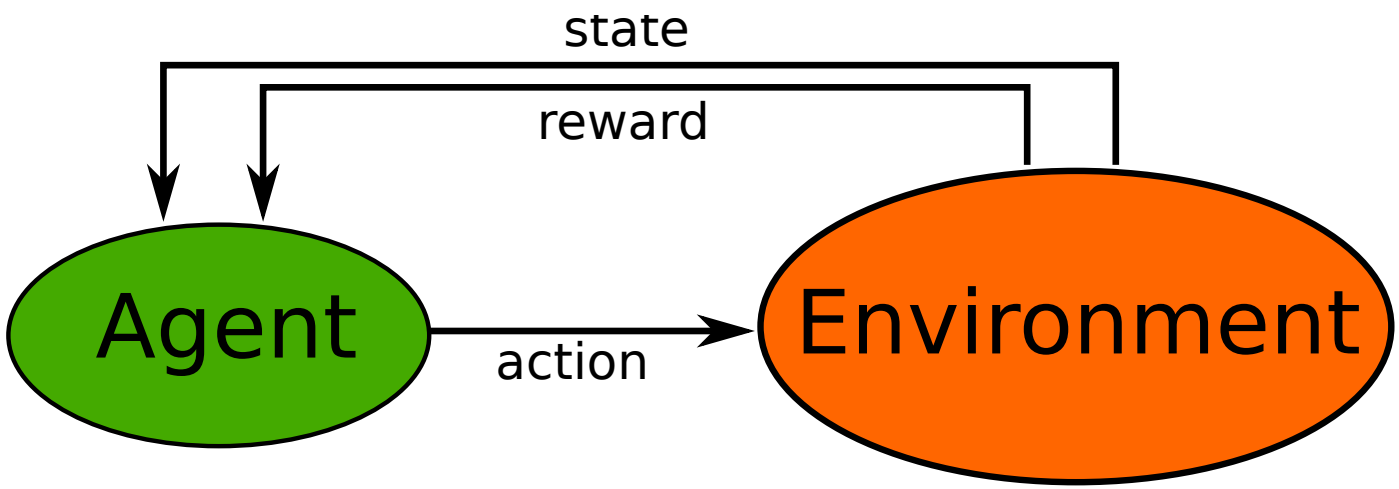
Reinforcement learning



We aim to model

- Systems where decisions are made in stages
- Immediate cost of current decision affects future costs
- Find decision making policies that minimize total cost

Terminology (Sutton view)



T: Find a mapping (policy) π from states s to actions a.
P: Maximize cumulative reward (return) $G = r_1 + r_2 + \ldots$
E: An RL data set looks as below:

$$\begin{aligned} D = &\{(s_1^{(1)}, a_1^{(1)}, r_1^{(1)}), (s_2^{(1)}, a_2^{(1)}, r_2^{(1)}), \ldots, (s_N^{(1)}, r_N^{(1)})\} \\ &\cup \{(s_1^{(2)}, a_1^{(2)}, r_1^{(2)}), (s_2^{(2)}, a_2^{(2)}, r_2^{(2)}), \ldots, (s_{N'}^{(2)}, r_{N'}^{(2)})\} \\ &\cup \ldots \end{aligned}$$

Episodic and continuous tasks

- State sequences of **episodic tasks** break naturally (i.e. a chess game):

$$S_1, A_1, R_2, S_2, A_2, R_3, S_3, A_3, R_4, S_4 = s_e \text{ (Episode 1)}$$
$$S_1, A_1, R_2, S_2, A_2, R_3, S_3, A_3, R_4, S_4, A_4, R_5, S_5 = s_e \text{ (Episode 2)}$$
$$\dots$$
$$S_1, A_1, R_2, S_2, A_2, R_3, S_3 = s_e \text{ (Episode M)}$$

  where s_e is the end state.
- **Continuous tasks** never end.

The Markov Property

Given a *good enough description* of present, the future is independent of the past:

$$P(S_{t+1}|S_t) = P(S_{t+1}|S_1, \cdots, S_t).$$

- Does not mean we do not care about the past.
- Means we can encapsulate it in *some* careful definition of state.
- Encourages effective state design.
- Greatly simplifies the system of random variables we need to tackle.

The Markov Process

A tuple of two entities $\langle \mathcal{S}, \mathcal{P} \rangle$, where

- $\mathcal{S}$ is the set of environment states.
- $\mathcal{P} = P(S_{t+1}|S_t)$ is the **environment dynamics model**.
  Also known as the *transition probability distribution*.
  I will call it the *transition model*.

The Markov Reward Process

A tuple of four entities $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where

- $\mathcal{S}$ is the set of environment states: $S_t = s$ with $s \in \mathcal{S},\ \forall t$.
- $\mathcal{R}$ is the set of rewards: $R_t = r$ with $r \in \mathcal{R},\ \forall r$.
- $\gamma \in [0, 1]$ is the **discount factor**.
- $\mathcal{P} = P(R_{t+1}, S_{t+1}|S_t)$ is the **environment dynamics model** that naturally decomposes according to the chain rule as

$$P(R_{t+1}, S_{t+1}|S_t) = \underbrace{P(R_{t+1}|S_{t+1}, S_t)}_{Reward\ model} \times \underbrace{P(S_{t+1}|S_t)}_{transition\ model} .$$

We keep the assumption that the state transitions follow the Markov property

$$P(S_{t+1}|S_t) = P(S_{t+1}|S_1, \cdots, S_t).$$

Random variables of an episode

Take the episode below:

$$S_1, R_2, S_2, R_3, S_3$$

How would its joint distribution decompose?

Random variables of an episode

First follow the chain rule (this time in probability theory, not calculus):

$$
\begin{aligned}
&P(S_1, R_2, S_2, R_3, S_3) \\
&\quad = P(S_3, R_3, S_2, R_2 | S_1) P(S_1) \\
&\quad = P(S_3, R_3, S_2 | R_2, S_1) P(R_2 | S_1) P(S_1) \\
&\quad = P(S_3, R_3 | S_2, R_2, S_1) P(S_2 | R_2, S_1) P(R_2 | S_1) P(S_1) \\
&\quad = \underbrace{P(S_3 | R_3, S_2, R_2, S_1)}_{P(S_3|S_2)} \underbrace{P(R_3 | S_2, R_2, S_1)}_{P(R_3|S_2)} \underbrace{P(S_2 | R_2, S_1)}_{P(S_2|S_1)} \\
&\qquad P(R_2 | S_1) P(S_1). \\
&\quad = P(S_3 | S_2) P(R_3 | S_2) P(S_2 | S_1) P(R_2 | S_1) P(S_1).
\end{aligned}
$$

Red: Markov property.

Blue: Reward model.

Return

Cumulative discounted reward starting from timestep t on

$$G_t \triangleq R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

- $\gamma^k R_{t+k+1}$ is the *present* value of the *future* reward R_{t+k+1}.
- $\gamma = 0$: Myopic return model
- $\gamma = 1$: Far-sighted return model

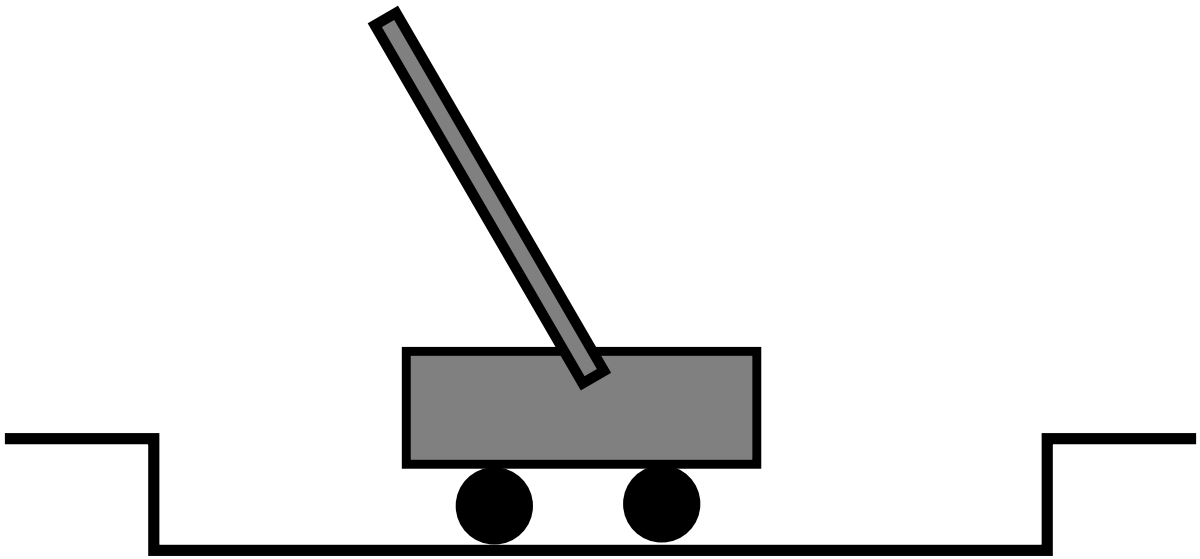
Why should we discount reward?

Account for two main risk factors:

- The environment model is not perfect.
  - Near future can be predicted more accurately than far future.
  - Make decisions based more on certain knowledge and less on uncertain knowledge (but based still on both).
- The learning model is not perfect.
  - We are up to *learning to decide*. The model itself will remain largely imperfect along the way.
  - How much would you rely on the advice of a child about how to invest your money for profit to come 20 years later?
    (what if it were an adult?)

Example: Pole balancing

Goal: Keep the cart on the track and the pole hinged on it away from falling down.



Can be modeled in both ways:

- **Episodic:** +1 reward per time step without failure.
- **Continuous:** -1 discounted reward for failure, 0 otherwise.

Recursiveness of Return

$$
\begin{aligned}
G_t &\triangleq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \\
&\triangleq R_{t+1} + \gamma \underbrace{\left[R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots \right]}_{G_{t+1}} \\
&\triangleq R_{t+1} + \gamma G_{t+1}.
\end{aligned}
$$

Makes the divide-and-conquer strategy applicable to RL.

The state-value function

$$v(s) = \mathbb{E}[G_t | S_t = s]$$

Where does $\mathbb{E}[\cdot]$ originate from? What is the source of stochasticity here?

The state-value function

$$v(s) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots]$$

- Expectation needs to be taken over all random variables

$$S_1, S_2, \cdots, R_1, R_2, \cdots$$

- The problem is that we have infinitely many of them!
  - **Markov:** State transitions follow the Markov property.
  - **Reward:** We model rewards as random variables.
  - **Process:** We have a collection of potentially unlimited set of random variables (i.e. a stochastic process).

The state-value function

$$v(s) = \sum_{S_t} \sum_{S_{t+1}} \cdots \sum_{R_{t+1}} \sum_{R_{t+2}} \cdots \Big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \Big]$$

The Bellman Equation

Shows how the recursiveness of return can be manipulated.

$$\begin{aligned} v(s) &= \mathbb{E}[G_t|S_t = s] \\ &= \mathbb{E}[R_{t+1} + \gamma G_{t+1}|S_t = s] \\ &= \underbrace{\mathbb{E}[R_{t+1}|S_t = s]}_{\langle r_s \rangle} + \gamma \mathbb{E}[G_{t+1}|S_t = s]. \end{aligned}$$

Let us take a closer look at the expectation in the second term

$$\mathbb{E}[G_{t+1}|S_t = s] = \sum_{s' \in \mathcal{S}} P[S_{t+1} = s'|S_t = s] \underbrace{\mathbb{E}[G_{t+1}|S_t = s, S_{t+1} = s']}_{v(s')}.$$

Then we get

$$v(s) = \mathbb{E}[R_{t+1}|S_t] + \gamma \sum_{s' \in \mathcal{S}} \underbrace{P[S_{t+1} = s'|S_t = s]}_{P_{ss'}} v(s').$$

The vectorized Bellman Equation

Let us repeat the Bellman equation for all possible states and store the outcomes into a vector

$$\mathbf{v} = \begin{bmatrix} v(s=1) \\ v(s=2) \\ \vdots \\ v(s=n) \end{bmatrix} = \begin{bmatrix} \langle r_1 \rangle + \gamma \sum_{s' \in \mathcal{S}} P_{1s'} v(s') \\ \langle r_2 \rangle + \gamma \sum_{s' \in \mathcal{S}} P_{2s'} v(s') \\ \vdots \\ \langle r_n \rangle + \gamma \sum_{s' \in \mathcal{S}} P_{ns'} v(s') \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} \langle r_1 \rangle \\ \langle r_2 \rangle \\ \vdots \\ \langle r_n \rangle \end{bmatrix}}_{\langle \mathbf{r} \rangle} + \gamma \underbrace{\begin{bmatrix} \sum_{s' \in \mathcal{S}} P_{1s'} v(s') \\ \sum_{s' \in \mathcal{S}} P_{2s'} v(s') \\ \vdots \\ \sum_{s' \in \mathcal{S}} P_{ns'} v(s') \end{bmatrix}}_{\mathbf{Pv}}$$

$$= \langle \mathbf{r} \rangle + \gamma \mathbf{P} \mathbf{v},$$

where $\mathbf{P}$ is the transition matrix with $\mathbf{P}[s, s'] = P_{ss'}$.

Solving the Bellman Equation for the Value Function

$$\begin{aligned} \mathbf{v} &= \langle \mathbf{r} \rangle + \gamma \mathbf{P} \mathbf{v} \\ \mathbf{v}(\mathbf{I} - \gamma \mathbf{P}) &= \langle \mathbf{r} \rangle \\ \mathbf{v} &= (\mathbf{I} - \gamma \mathbf{P})^{-1} \langle \mathbf{r} \rangle \end{aligned}$$

- Has complexity $O(n^3)$ for n states.
- Hence needs to be approximated for many real-world applications.
- How to approximate is a large portion of the remaining course material!

The Markov Decision Process

A tuple of five entities $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where

- $\mathcal{S}$ is the set of environment states: $S_t = s$ with $s \in \mathcal{S}, \ \forall t$.
- $\mathcal{A}$ is the set of actions: $A_t = a$ with $a \in \mathcal{A}, \ \forall a$.
- $\mathcal{R}$ is the set of rewards: $R_t = r$ with $r \in \mathcal{R}, \ \forall r$.
- $\gamma \in [0, 1]$ is the **discount factor**.
- $\mathcal{P} = P(R_{t+1}, S_{t+1} | S_t, A_t)$ is the **environment dynamics model** that naturally decomposes according to the chain rule as

$$P(R_{t+1}, S_{t+1} | S_t, A_t) = \underbrace{P(R_{t+1} | S_t, A_t)}_{Reward\ model} \times \underbrace{P(S_{t+1} | S_t, A_t)}_{transition\ model}.$$

We keep the assumption that the state transitions follow the Markov property

$$P(S_{t+1} | S_t) = P(S_{t+1} | S_1, \cdots, S_t).$$

Policy

Mind that we thus far had a new random variable A_t without an assigned distribution. That distribution is the **policy**, which is defined as a mapping from states to actions

$$\pi(A_t|S_t) = P(A_t|S_t).$$

- MDP models the environment.
- Policy models the agent.
- Our primary concern will be *stationary* policies

$$A_t \sim \pi(\cdot|S_t), \forall t > 0,$$

which behave invariantly of time.

MDP as a MRP

We will typically have an MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ with an attached policy π. Integrate out all actions wrt the policy distribution

$$P^{\pi}_{s,s'} = \sum_{a \in \mathcal{A}} \pi(A_t = a | S_t = s) P(S_{t+1} = s' | S_t = s, A_t = a),$$
$$r^{\pi}_{s} = \sum_{a \in \mathcal{A}} \pi(A_t = a | S_t = s) P(R_{t+1} | S_t = s, A_t = a).$$

Similarly to the MRP case, we construct a transition matrix and a reward vector by evaluating r^{π}_{s} for all states and $P^{\pi}_{s,s'}$ for all state pairs:

$$\mathbf{P}^{\pi}[s, s'] = P^{\pi}_{s,s'}, \qquad \mathbf{r}^{\pi}[s] = r^{\pi}_{s}.$$

We finally attain an MRP with $\langle \mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}, \gamma \rangle$.