Обучение с подкреплением

Введение: MDP и постановка задачи

Что такое обучение с подкреплением

Обучение с подкреплением (Reinforcement Learning, RL) — обучение через *взаимодействие*:

- **агент** совершает *действия*;
- **среда** выдаёт *награду* (reward) и новое *состояние*;
- цель — максимизировать *суммарную* награду за всё время.

Что такое обучение с подкреплением

Обучение с подкреплением (Reinforcement Learning, RL) — обучение через *взаимодействие*:

- **агент** совершает *действия*;
- **среда** выдаёт *награду* (reward) и новое *состояние*;
- цель — максимизировать *суммарную* награду за всё время.

В отличие от supervised learning: нет «правильного ответа» для каждого входа — есть лишь скалярный сигнал о том, насколько хорошо действует агент.

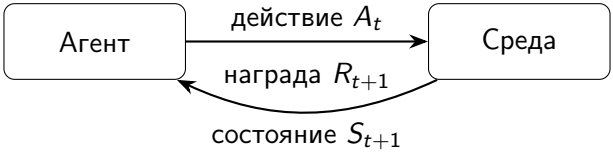
Три парадигмы обучения

- **Обучение с учителем.** Дана пара (x, y), учим $f : x \mapsto y$.
- **Обучение без учителя.** Меток нет: ищем структуру в данных.
- **Обучение с подкреплением.**
  - меток нет, но есть *скалярная награда*;
  - награда может быть *отложенной* (delayed reward);
  - агент сам влияет на данные, которые видит;
  - возникает проблема *exploration vs. exploitation*.

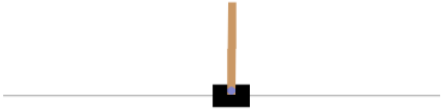
Взаимодействие агента со средой

Дискретные шаги $t = 0, 1, 2, \ldots$. На каждом шаге:

1. агент наблюдает состояние $S_t \in \mathcal{S}$;
2. выбирает действие $A_t \in \mathcal{A}$;
3. среда выдаёт награду $R_{t+1} \in \mathcal{R}$ и состояние S_{t+1}.



Среда 1: CartPole



Вопрос: что здесь является состоянием, действием и наградой?

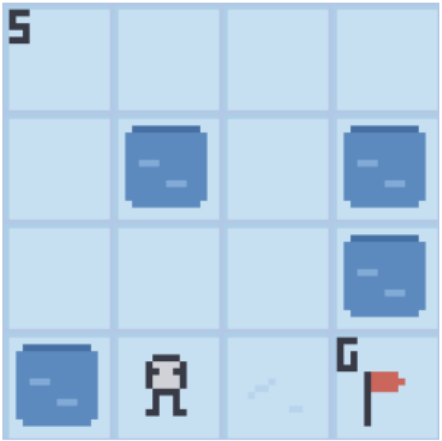
Ответ: CartPole

CartPole — стабилизация шеста на тележке.

- **Состояние**: $s = (x, \dot{x}, \theta, \dot{\theta})$;
- **Действие**: толкать влево/вправо, $A_t \in \{0, 1\}$;
- **Награда**: $+1$ за каждый шаг, пока шест не упал;
- **Завершение**: $|\theta|$ или $|x|$ превысили порог, либо лимит шагов.



Среда 2: FrozenLake

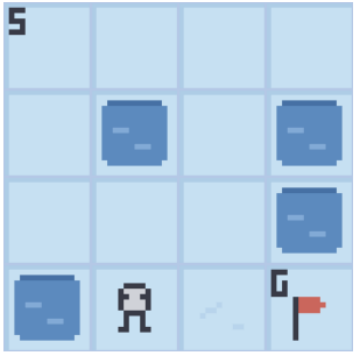


Вопрос: что здесь является состоянием, действием и наградой?

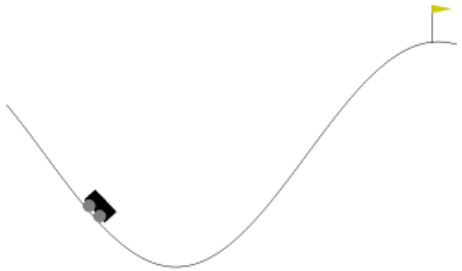
Ответ: FrozenLake

FrozenLake — прогулка по замёрзшему озеру.

- **Состояние**: клетка $0, \ldots, 15$ (дискретное);
- **Действие**: четыре направления;
- **Награда**: $+1$ при достижении цели G, 0 иначе;
- **Среда**: скользкий лёд (стохастические переходы).



Среда 3: MountainCar

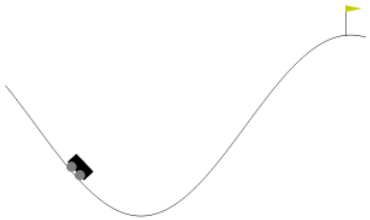


Вопрос: что здесь является состоянием, действием и наградой?

Ответ: MountainCar

MountainCar — машина в ложбине между холмами.

- **Состояние**: $s = (x, v)$ — положение и скорость;
- **Действие**: газ влево / ничего / газ вправо;
- **Награда**: -1 за шаг, пока вершина не достигнута;
- **Среда**: не хватает мощности — нужно сначала раскачаться.



Среда 4: Pendulum



Вопрос: что здесь является состоянием, действием и наградой?

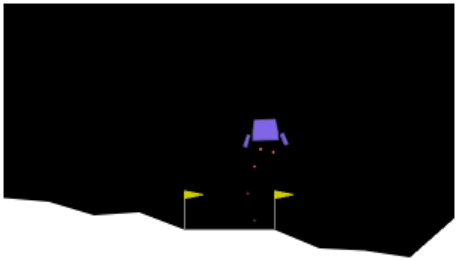
Ответ: Pendulum

Pendulum — перевёрнутый маятник.

- **Состояние**: $s = (\cos\theta, \sin\theta, \dot{\theta})$;
- **Действие**: крутящий момент $a \in [-2, 2]$ (непрерывное);
- **Награда**: отрицательная, зависит от отклонения и момента.



Среда 5: LunarLander

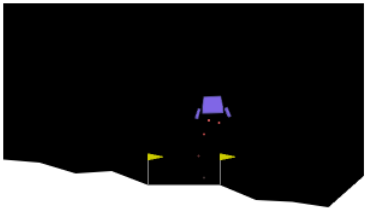


Вопрос: что здесь является состоянием, действием и наградой?

Ответ: LunarLander

LunarLander — посадка аппарата на Луну.

- **Состояние**: 8-мерный вектор;
- **Действие**: ничего / левый / главный / правый двигатель;
- **Награда**: за мягкую посадку, штрафы за топливо и жёсткую посадку.



Доход (return) и дисконтирование

Доход (return)

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

где $\gamma \in [0, 1]$ — коэффициент дисконтирования.

- $\gamma = 0$ — агент «близорук»;
- $\gamma \to 1$ — учитывает далёкое будущее;
- $\gamma < 1$ гарантирует конечность G_t при ограниченных наградах.

Цель обучения

Целевой функционал

$$J(\pi) = \mathbb{E}_\pi[G_0] = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1}\right].$$

Именно эту величину алгоритм максимизирует в ходе обучения:

$$\pi^* = \arg\max_\pi J(\pi).$$

На практике $J(\pi)$ оценивают по эпизодам — средний return по нескольким запускам.

Марковские процессы: цепочка усложнения

1. **Марковское свойство** — будущее зависит только от настоящего;
2. **MRP** — добавляем награды;
3. **MDP** — добавляем действия и политику.

$$\text{Markov chain} \xrightarrow{+R,\gamma} \text{MRP} \xrightarrow{+A,\pi} \text{MDP}$$

Марковское свойство

$$P(S_{t+1} \mid S_t, A_t) = P(S_{t+1} \mid S_t, A_t, S_{t-1}, A_{t-1}, \ldots, S_0, A_0).$$

Марковские среды: GridWorld, FrozenLake, шахматы, CartPole (с полным наблюдением).

MRP и MDP

MRP — четвёрка $(\mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma)$:

$$\mathcal{P}_{ss'} = P(S_{t+1} = s' \mid S_t = s), \qquad \mathcal{R}_s = \mathbb{E}[R_{t+1} \mid S_t = s].$$

MDP — кортеж $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:

$$\mathcal{P}^a_{ss'} = P(S_{t+1} = s' \mid S_t = s, A_t = a), \qquad \mathcal{R}^a_s = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a].$$

MDP — основной формализм RL: добавляется политика агента.

Если марковость нарушена (POMDP)

Наблюдается не полное состояние, а наблюдение o_t — частично наблюдаемый MDP.
Что помогает:

- **frame stacking** — стек последних k наблюдений (Atari);
- **рекуррентные агенты** (LSTM/GRU) — сеть хранит историю;
- **belief state** — распределение по истинному состоянию (POMDP-методы).

RL применим и при частичной наблюдаемости, но с оговорками.

Политика

$$\pi(a \mid s) = P(A_t = a \mid S_t = s).$$

- стохастическая политика — распределение по действиям;
- детерминированная — частный случай: $\pi(a \mid s) \in \{0, 1\}$.

Функции ценности

Ценность состояния

$$V^{\pi}(s) = \mathbb{E}_{\pi}[G_t \mid S_t = s].$$

Ценность действия (Q-функция)

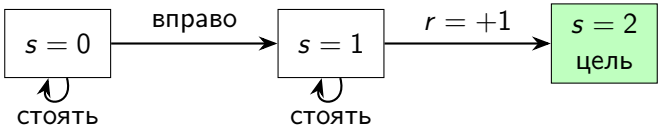
$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}[G_t \mid S_t = s,\ A_t = a].$$

Связь:

$$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s)\, Q^{\pi}(s, a).$$

Пример: как работают V и Q

Полоска клеток $0, 1, 2$; цель в клетке 2 (награда $+1$). Действия: «вправо», «стоять». Детерминированные переходы, $\gamma = 0{,}9$, политика π — всегда «вправо».



$$V^\pi(2) = 0, \qquad V^\pi(1) = 1, \qquad V^\pi(0) = 0{,}9.$$

Пример: Q-значения

Связь Q и V для детерминированных переходов:

$$Q^\pi(s,a) = r(s,a) + \gamma\, V^\pi(s'), \qquad s' = f(s,a).$$

$$Q^\pi(1, \text{вправо}) = 1, \qquad Q^\pi(1, \text{стоять}) = 0{,}9,$$
$$Q^\pi(0, \text{вправо}) = 0{,}9, \qquad Q^\pi(0, \text{стоять}) = 0{,}81.$$

В каждой клетке «вправо» лучше: жадная политика $\arg\max_a Q^\pi(s,a)$ совпадает с π.

Пример: выбор лучшей политики

Качество политики: $J(\pi) = V^{\pi}(0)$ (старт всегда в клетке 0).

- π_1: всегда «вправо» $\Rightarrow J(\pi_1) = 0{,}9$
- π_2: всегда «стоять» $\Rightarrow J(\pi_2) = 0$
- π_3: «вправо» в 0, «стоять» в 1 $\Rightarrow J(\pi_3) = 0$
- π_4: стохастическая (0,8 вправо / 0,2 стоять) $\Rightarrow J(\pi_4) \approx 0{,}857$

$$J(\pi_1) = 0{,}9 > J(\pi_4) \approx 0{,}857 > J(\pi_2) = J(\pi_3) = 0.$$

Лучшая — π_1: только она стабильно доходит до цели и забирает награду.

On-policy и off-policy

Off-policy — может обучаться на опыте *произвольной* стратегии.
On-policy — для каждой итерации нужен опыт *своей* стратегии.

- target policy π — целевая политика;
- behavior policy μ — политика сбора данных;
- off-policy учится «не по себе», on-policy «привязан» к себе;
- on-policy — на своих ошибках, off-policy — и на своих, и на чужих.

Основные подходы

- **Value-based** — оптимальная политика через функции ценности (DQN);
- **Policy gradient** — градиент $J(\pi)$ по параметрам политики (REINFORCE, A2C, PPO);

Критерии оценки RL-алгоритмов

- **Performance** — Монте-Карло оценка $J(\pi)$:

$$\hat{J}(\pi) = \frac{1}{N}\sum_{i=1}^{N} G_0^{(i)}, \qquad G_0^{(i)} = \sum_{t=0}^{T_i} \gamma^t r_{t+1}^{(i)}.$$

- **Wall-clock time** — реальное время работы;
- **Sample efficiency** — число шагов среды $N_{\text{env}} = \sum_{i=1}^{N} T_i$ (ключевое, если среда дорогая).

Критерии часто конфликтуют — сравнивать нужно по всем трём осям.

Итоги

- RL — обучение через взаимодействие; цель — максимизировать G_t.
- $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$.
- MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ — формальная модель.
- $V^\pi(s)$ и $Q^\pi(s, a)$ — функции ценности.