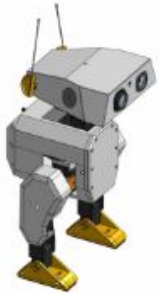
ЦЕНТР МОЛОДЁЖНОЙ
РОБОТОТЕХНИКИ
МГТУ ИМ. Н.Э. БАУМАНА

Основы reinforcement learning. RL-алгоритмы.

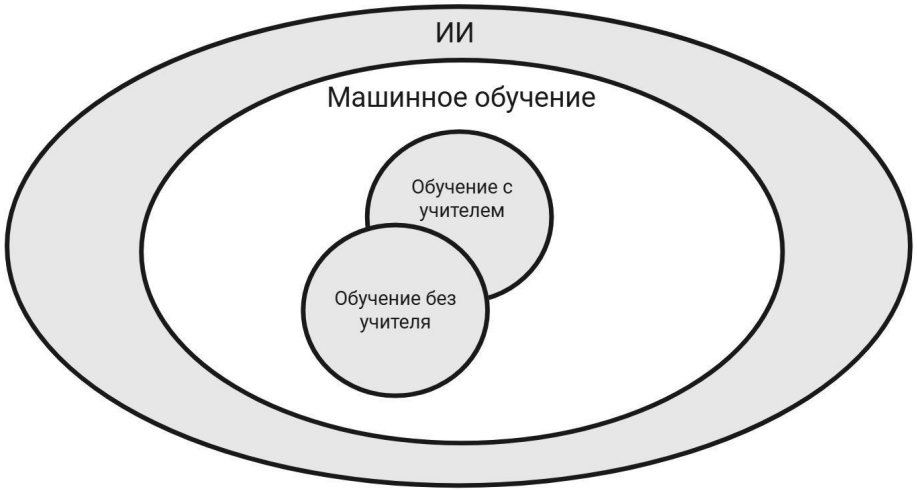
(1) Manipulation

(2) Locomotion



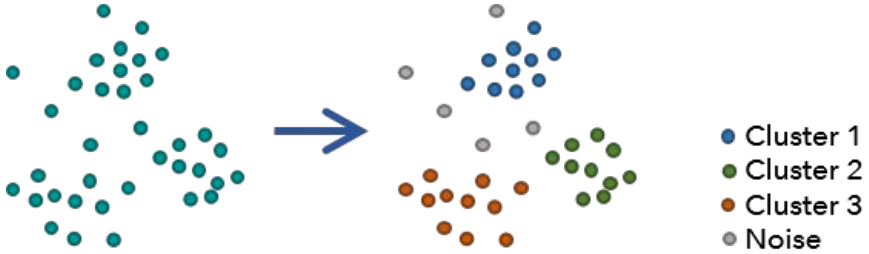
(3) Whole-body control

Основные направления МО

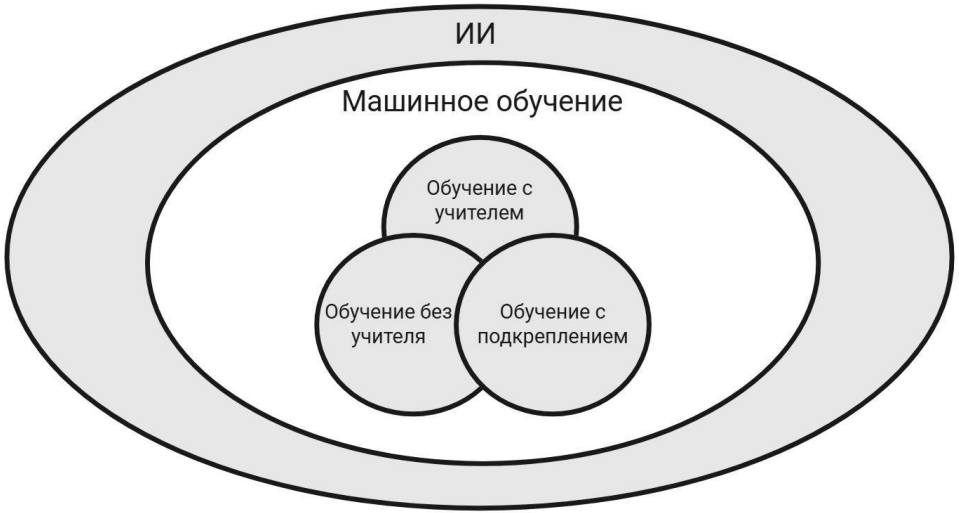


Обучение с учителем (supervised learning) – построение алгоритмов неявным образом за счет взаимодействия с **размеченной выборкой**

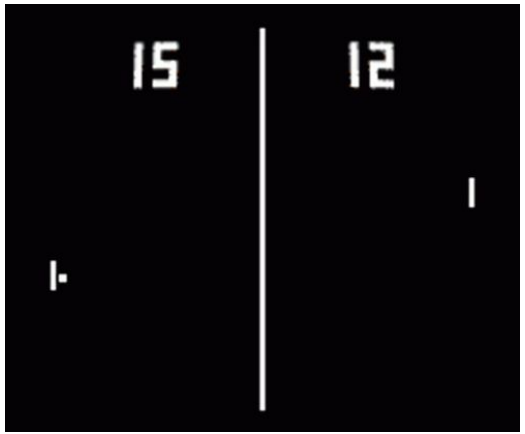
Обучение без учителя (unsupervised learning) – построение алгоритмов неявным образом за счет взаимодействия с **неразмеченной выборкой**

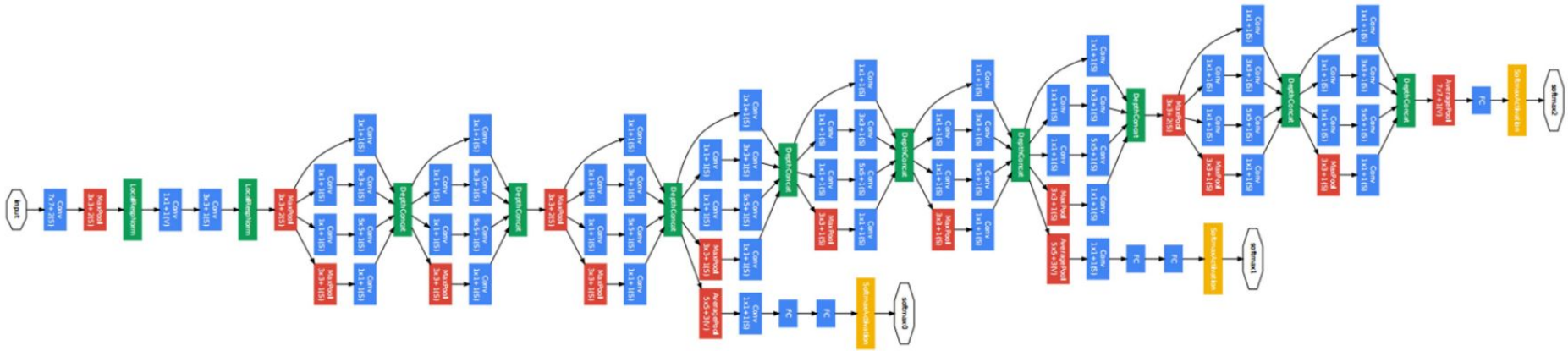


Основные направления МО



Обучение с подкреплением - построение алгоритмов неявным образом за счет взаимодействия со средой **методом проб и ошибок**

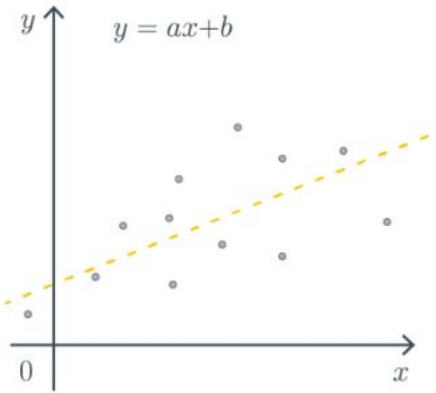




$$f(x) = y$$

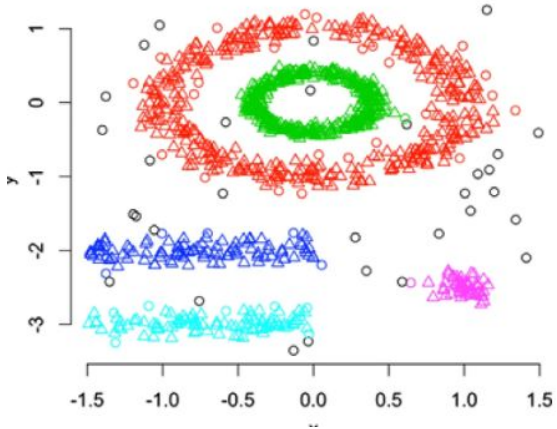
Обучение с учителем

Модель делает прогноз на основе размеченных данных



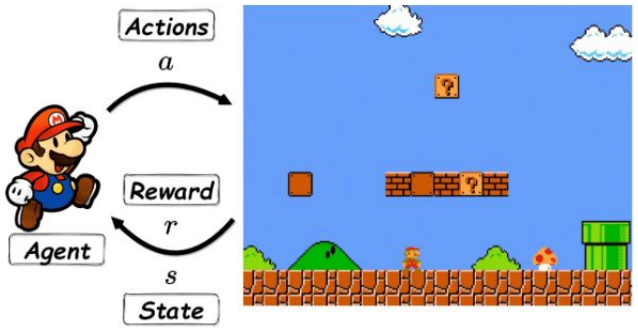
Обучение без учителя

Модель напрямую не взаимодействует с правильными ответами

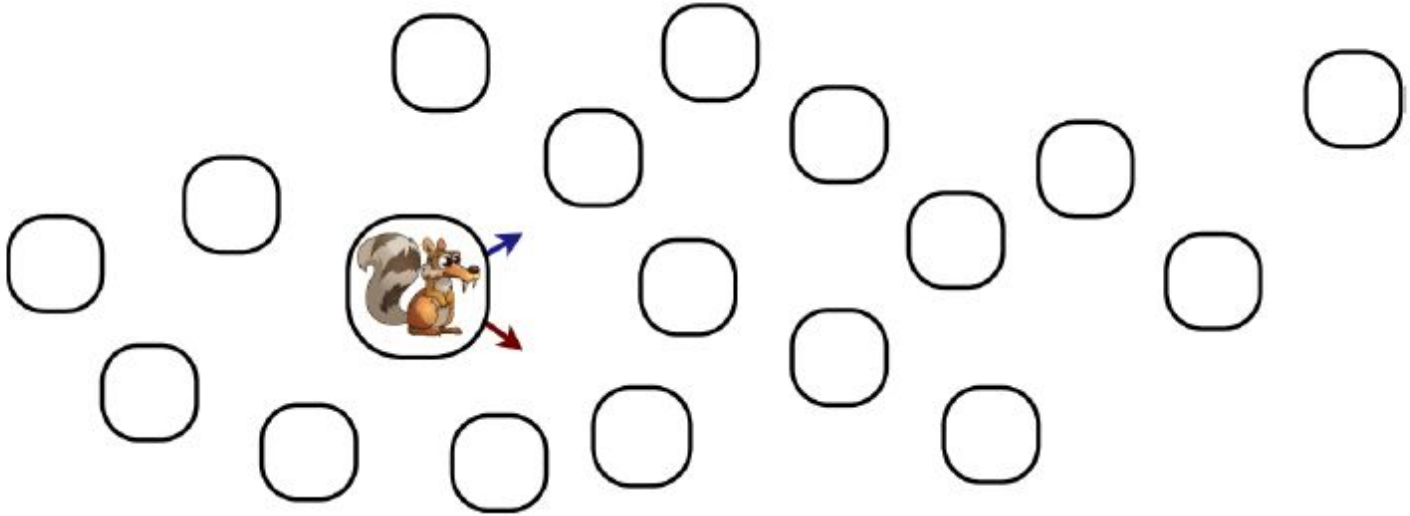


Обучение с подкреплением

Модель получает награду за правильное действие на основании строгих критериев



Агент и среда



Агент (Agent) = Контроллер (код, принимающий решение, что делать дальше)
Среда (Environment) = Всё остальное:

- "Железо" (сервомоторы, двигатели, клешня, лапы, ноги и так далее),
- Физический симулятор,
- Объекты в зоне досягаемости (или другой агент)

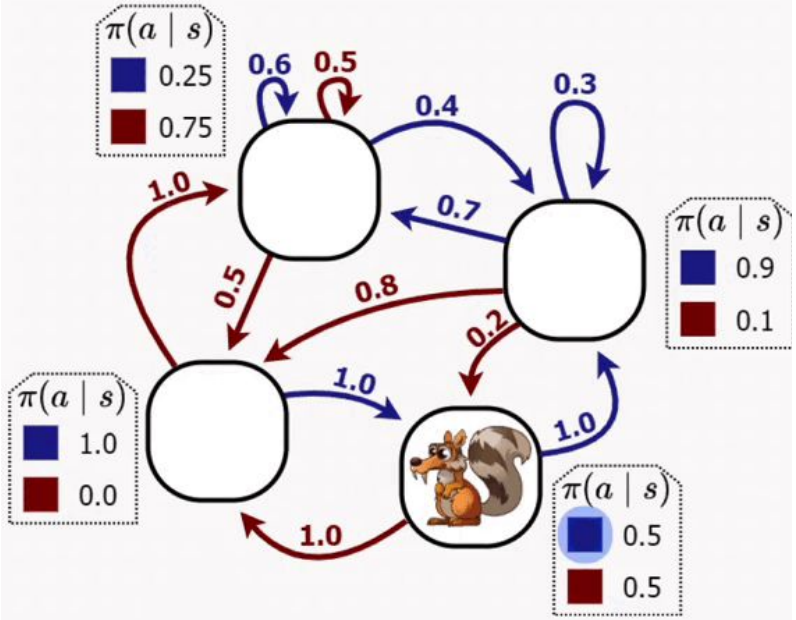
Стратегия (policy)

Задача найти стратегию (policy) π, максимизирующую среднюю награду:

$$\mathbb{E}_{\mathcal{T}\sim\pi}\sum_{t>0} r_t \to \max_{\pi},$$

Награда за стратегию:

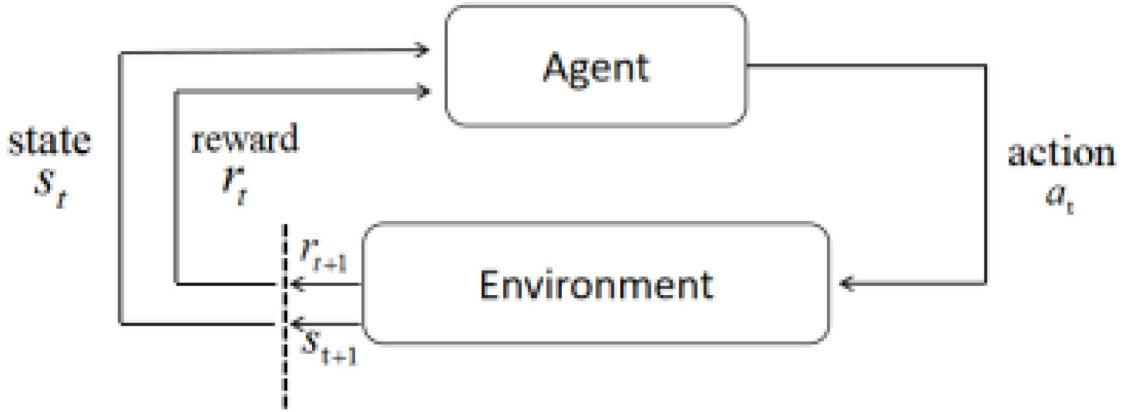
$$J(\pi) := \mathbb{E}_{\mathcal{T}\sim\pi} R(\mathcal{T})$$

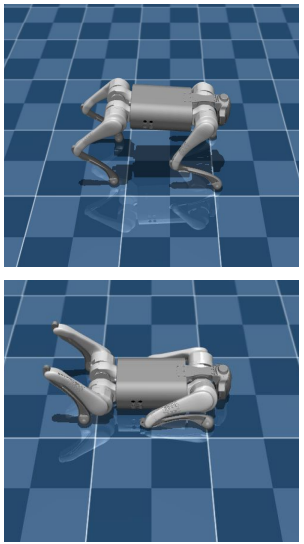


Дисконтирование награды – домножение суммарной награды на коэффициент для учета долгосрочной выгоды

Коэффициент дисконтирования: $\gamma \in [0..1]$

Итеративность обучения





- **Эпизод** – цикл от начального состояния до терминального
- **Терминальное состояние** – состояние завершения эпизода

Агент входит в терминальное состояние в случае:

Успеха: - Достижения цели (целевого состояния)

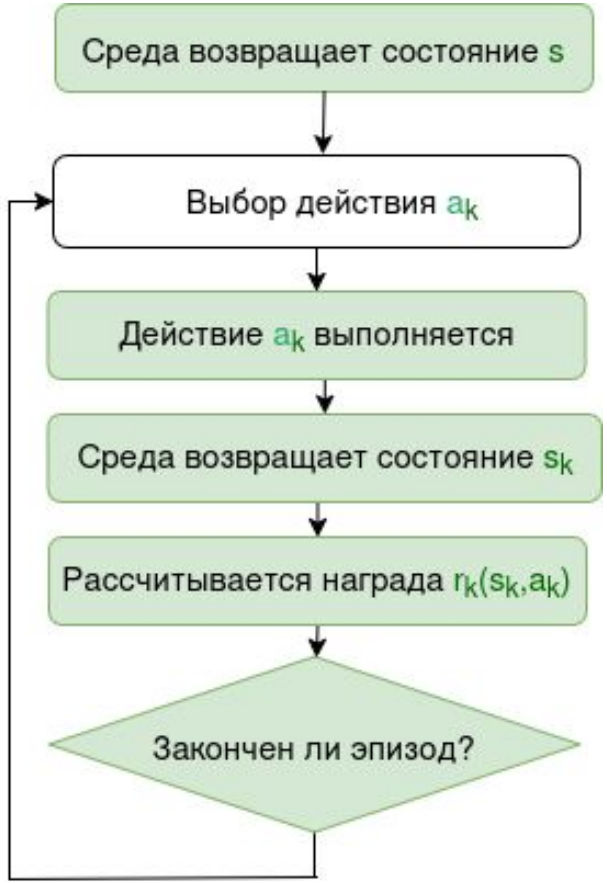
Неудачи: - Истечения времени на эпизод
- Нарушения правил (падение шагающего робота, выход за границы, ...)

Цикл RL-агента

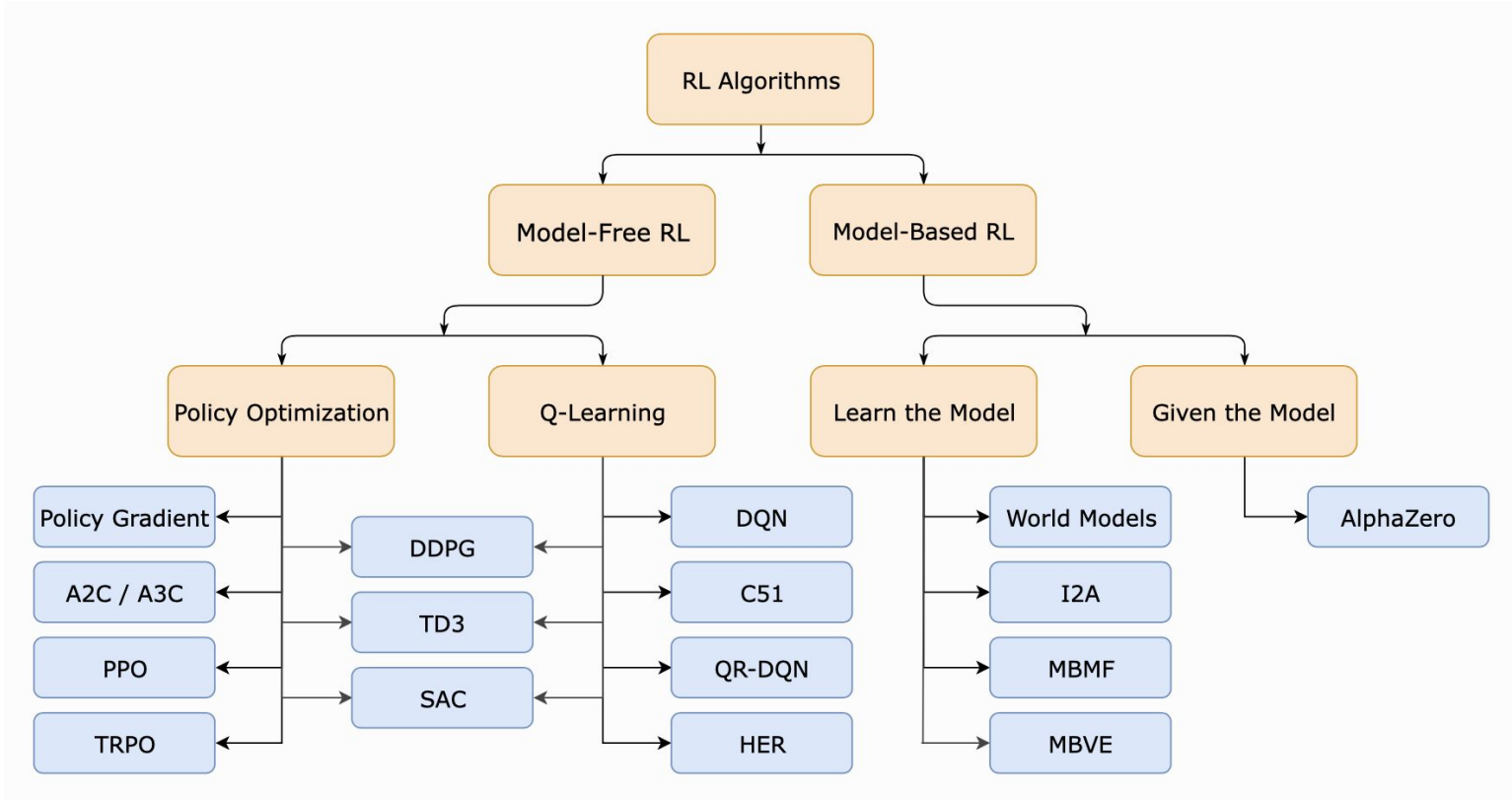
Необходимо определить стратегию для выбора оптимальных действий — таких, чтобы достичь максимальной награды

Основная терминология:

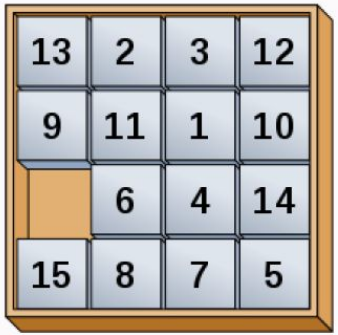
- Состояние (State)
- Действие (Action)
- Награда (Reward)
- Политика (Policy)
- Эпизод (Episode)



Виды RL-алгоритмов

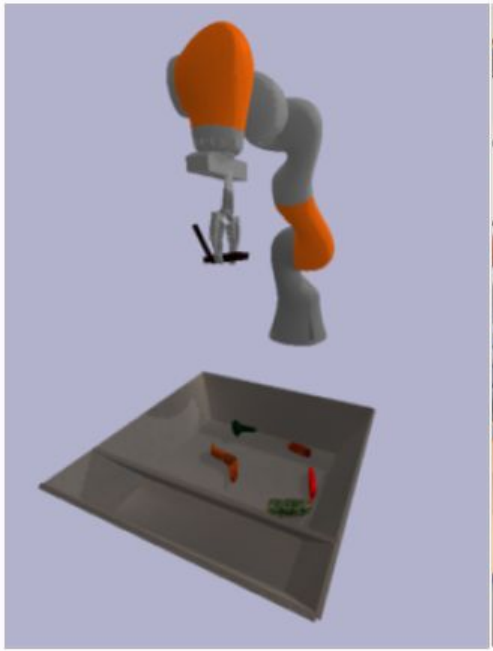


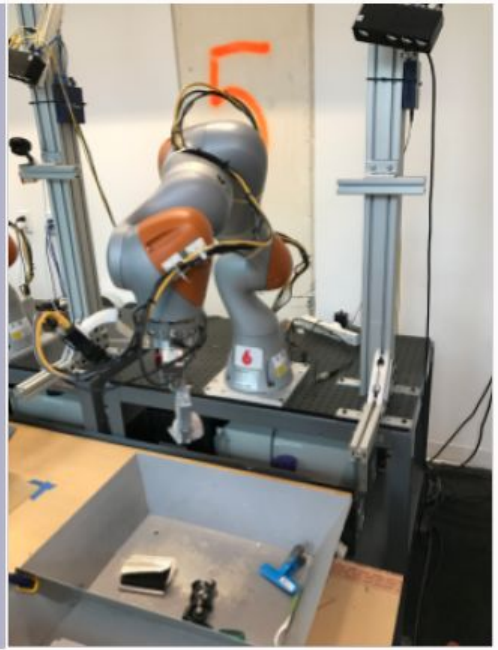
Model-based



Знаем и учим будущее состояние

Model-free





Информация о динамике среды закрыта

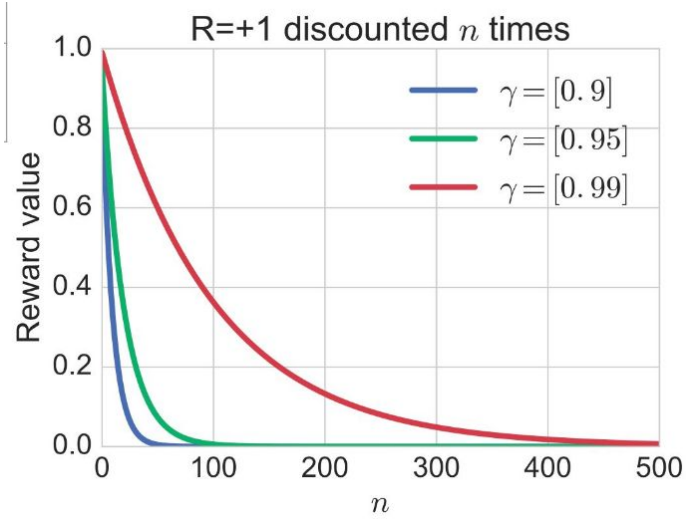
Оценочная функция

$$G_0 = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}$$

Сумма всех будущих наград.

Оценочная функция показывает будущую награду, если в текущем состоянии s выбрать действие a.

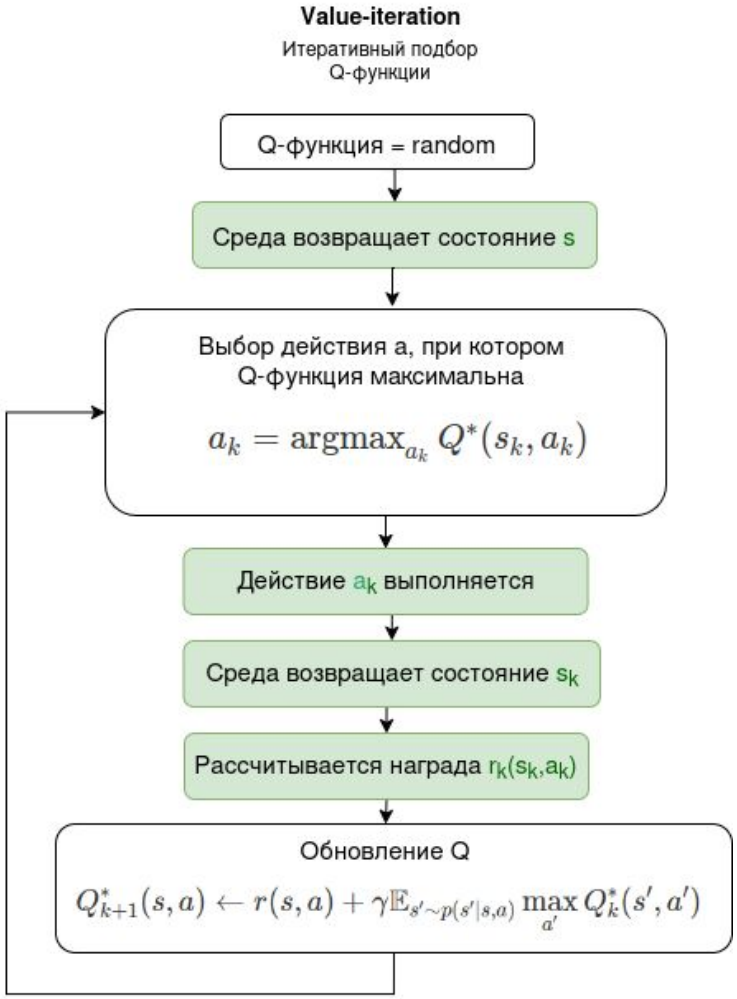
Вклад отдалённых наград меньше, т. к. награды умножаются на коэффициент дисконтирования.



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

discount factor

$\gamma \in [0..1]$



Value-iteration

Model-based алгоритм

Подбираем **истинную** Q-функцию.

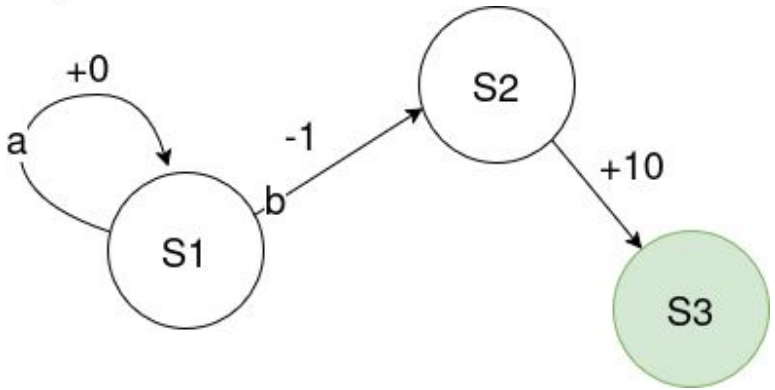
Алгоритм динамического программирования для поиска оптимальной стратегии для марковского процесса принятия решений (MDP) путём итеративного обновления значений каждого состояния.

Value-iteration

0. Инициализация

$$V_0(S1) = V_0(S2) = V_0(S3) = 0$$

$$\gamma = 0.9$$



4. Результат

$$V^*(S1) = 8, \quad V^*(S2) = 10, \quad V^*(S3) = 0$$

1 Итерация

S2: $Q(S2, b) = 10 + 0.9 \cdot V_0(S3) = 10 + 0 = 10 \rightarrow V_1(S2) = 10$

S1: $Q(S1, a) = 0 + 0.9 \cdot V_0(S1) = 0$

$Q(S1, b) = -1 + 0.9 \cdot V_0(S2) = -1 + 0 = -1$

$\rightarrow V_1(S1) = \max(0, -1) = 0$

2 Итерация

S2: $Q(S2, b) = 10 + 0.9 \cdot V_1(S_3) = 10 \rightarrow V_2(S_2) = 10$

S1: $Q(S1, a) = 0 + 0.9 \cdot V_1(S1) = 0$

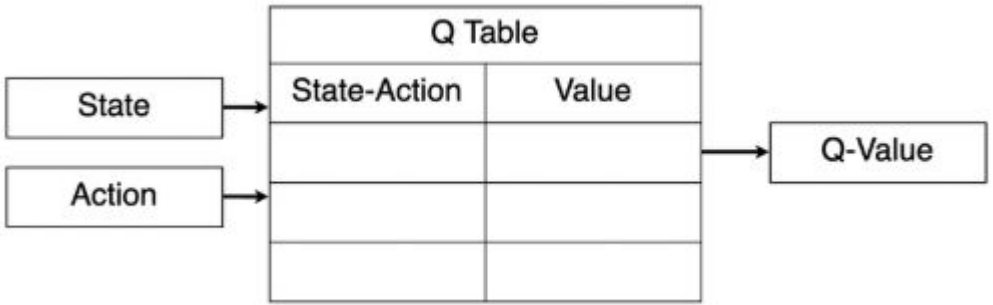
$Q(S1, b) = -1 + 0.9 \cdot V_1(S2) = -1 + 9 = 8$

$\rightarrow V_2(S1) = \max(0, 8) = 0$

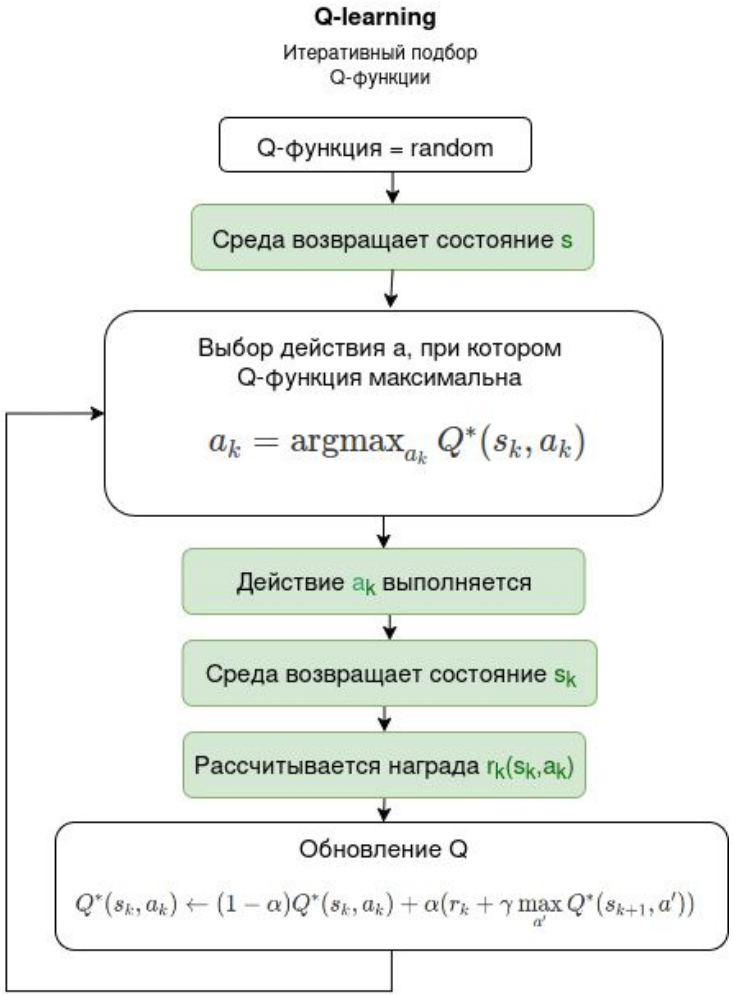
Q-learning

Агент от среды получает награды, на основании которой может **сформировать** оценочную функцию Q.

Функция позволяет без модели окружающей среды, оценивать полезность доступных действий.



$$Q^{new}(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}.$$

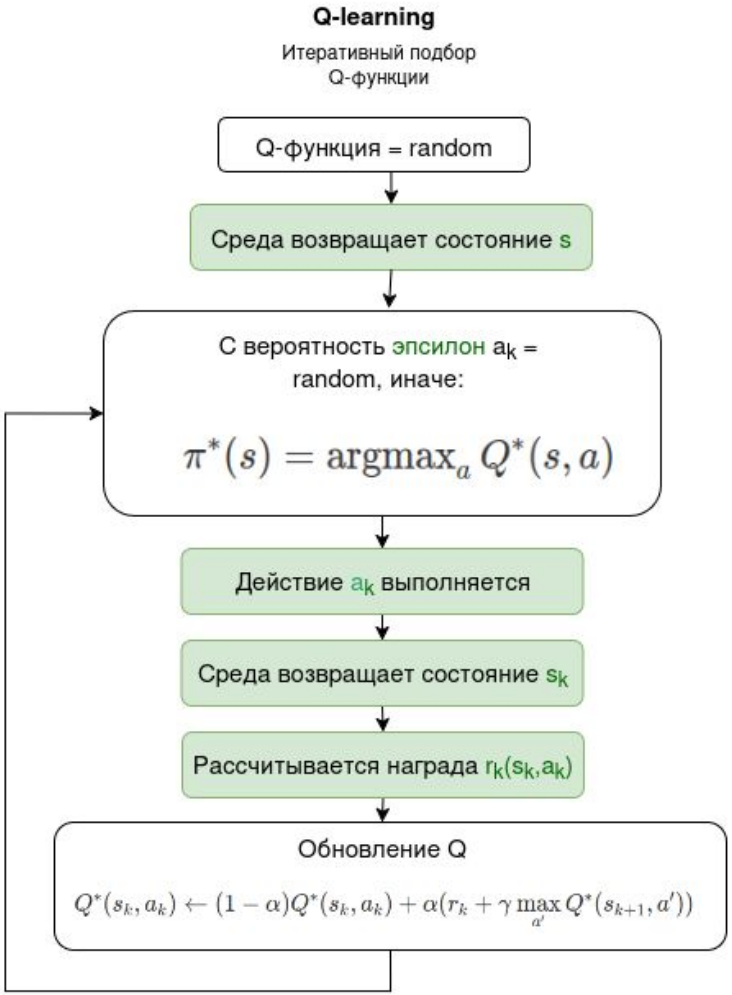


Q-learning

Model-free алгоритм

Точная матрица переходов неизвестна, а значит неизвестны и точные будущие вознаграждения.

Подбираем **примерную** Q-функцию (стохастичную оценку)



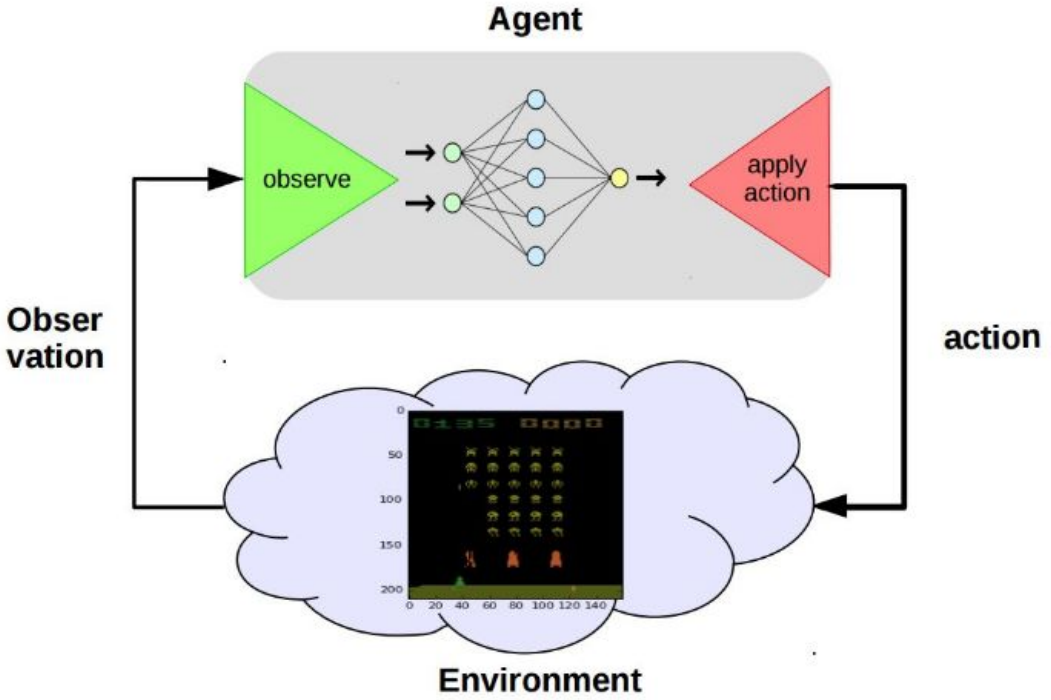
Q-learning

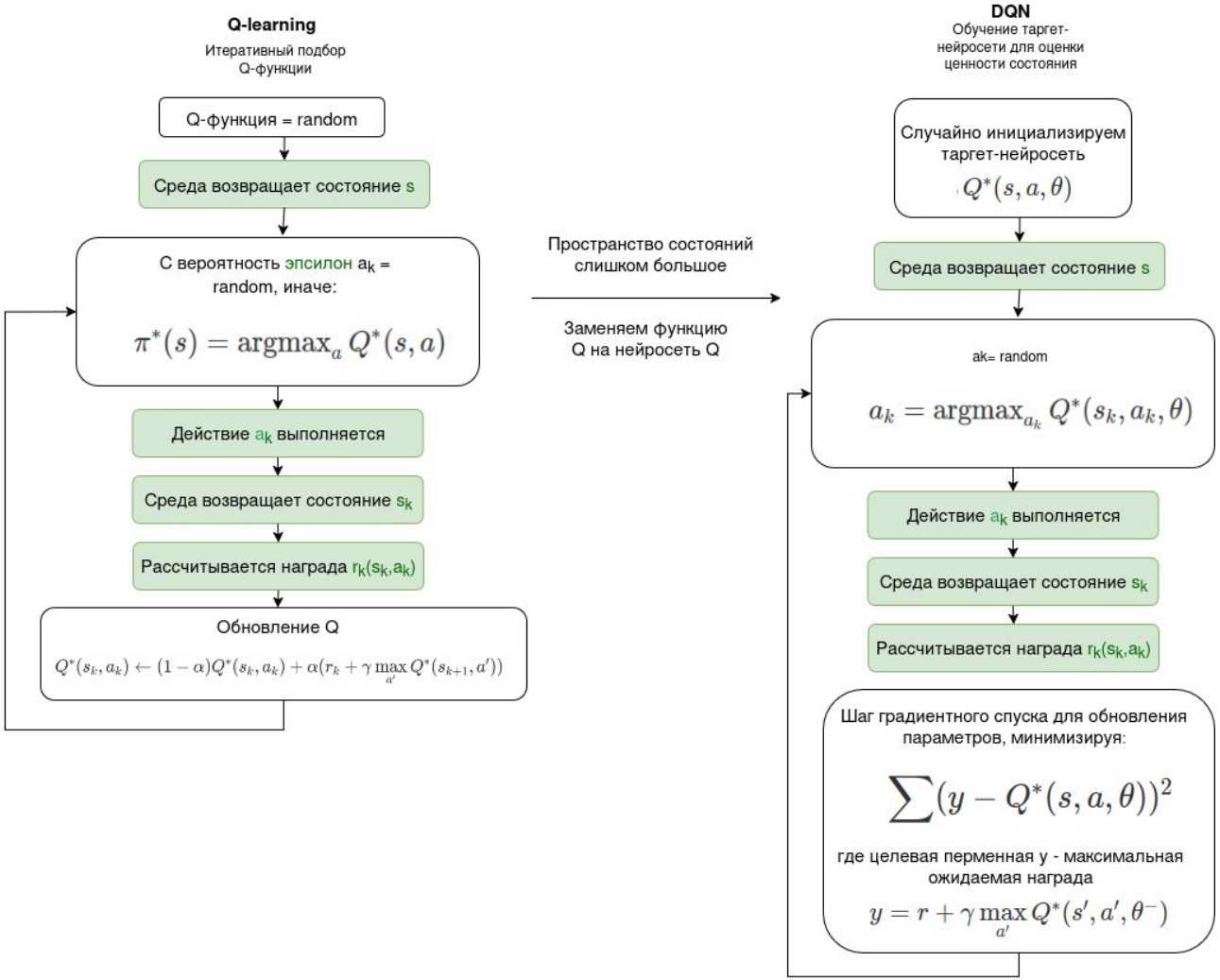
С вероятность эпсилон выбираем не самые оптимальные действия, а случайные.

Развитие идеи Q-learning

Если пространство состояний слишком большое, уже не получится хранить Q как таблицу в памяти

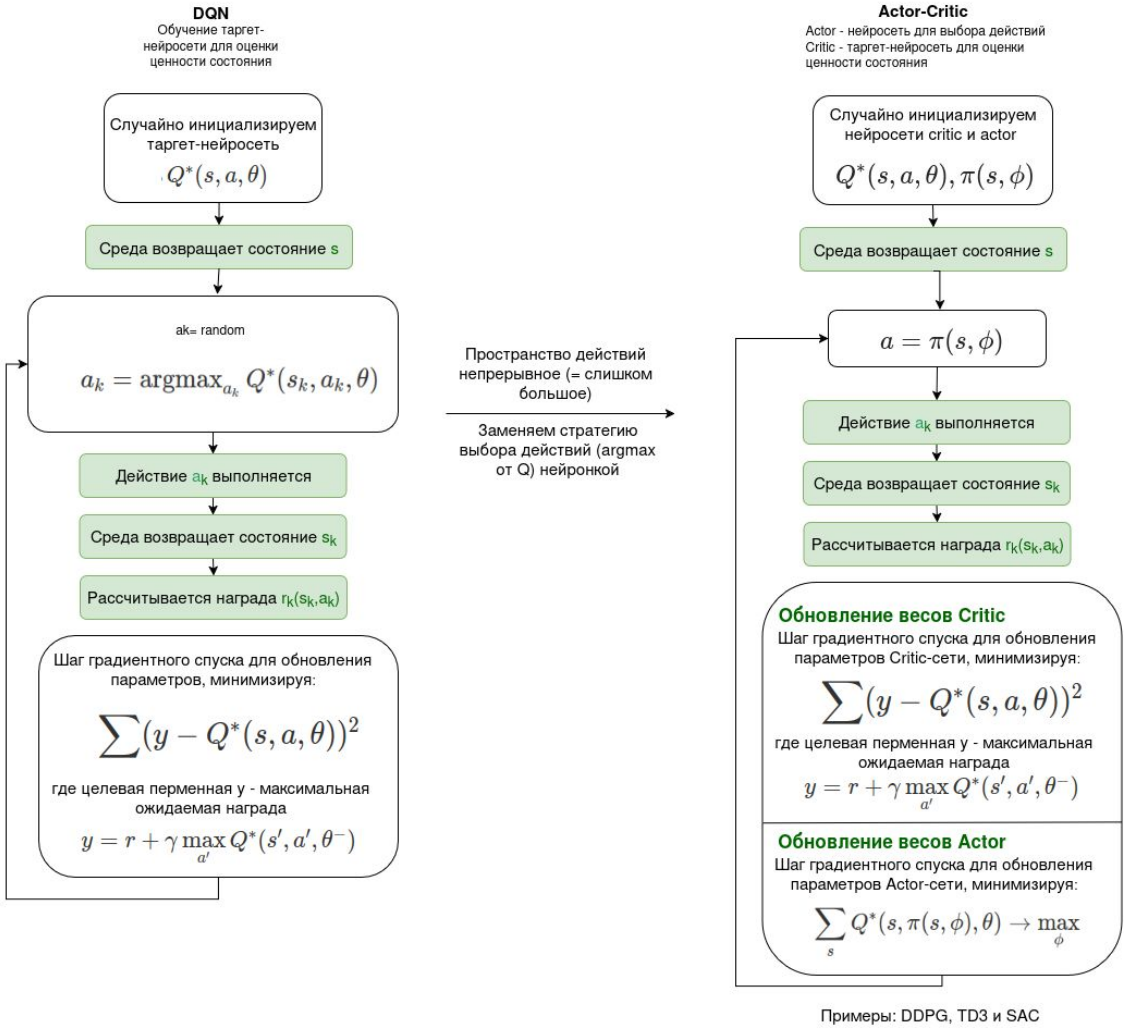
Тогда можно заменить Q-функцию на Q-нейросеть





DQN

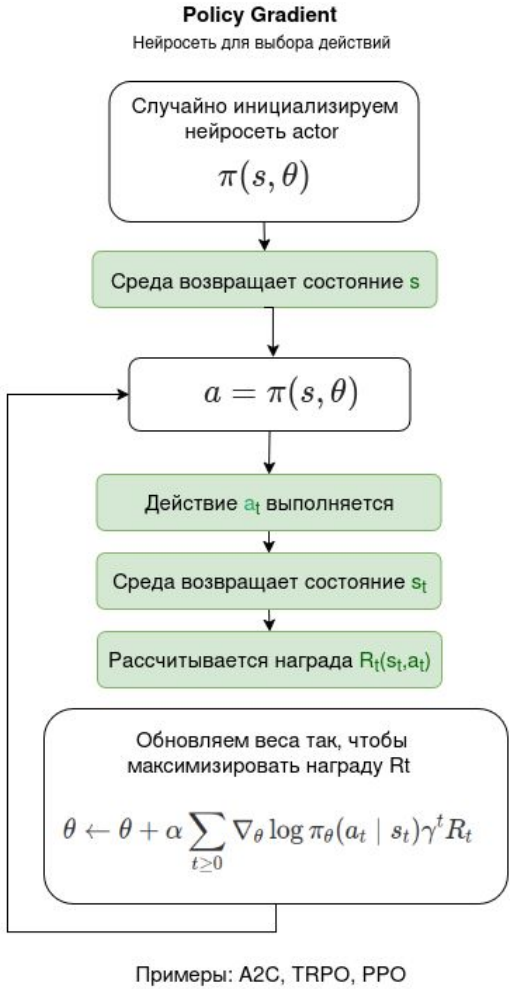
Вместо функции ценности Q — таргет-нейросеть для оценки ценности



Actor-Critic

-Вместо функции ценности Q — Critic-нейросеть для оценки ценности

-Вместо эпсилон-жадного выбора — Actor-нейросеть для предсказания действий



Policy Gradient

On-policy алгоритм

Напрямую оптимизирует то, что нам нужно: поведение агента (а не оценочную функцию).

Сбор данных в model-free алгоритмах

- (off-policy) Value-based подход подразумевает, что алгоритм ищет не саму стратегию, а оптимальную Q-функцию

Примеры: DDPG, TD3 и SAC

- (on-policy) Policy Gradient – градиенты по стратегиям. Возвращается вероятностное распределение действий, которое задается выходами нейронной сети.

Примеры: Advantage Actor-Critic (A2C), Trust-Region Policy Optimization (TRPO) и Proximal Policy Optimization (PPO)