

Методы искусственного интеллекта

Лекция 7. Обучение с подкреплением

Тема 14. Исследование или использование (exploration vs. exploitation). Многорукие бандиты

Особенности задачи

- До этого:
 - Есть модель задачи:
 - Детерминированная (поиск)
 - Вероятностная (байесовские сети)
 - Есть набор примеров:
 - Обучение с учителем
 - Нейронные сети и пр.
- Сейчас:
 - Нет модели
 - Нет данных
 - Есть среда (окружение) и набор действий, которые в нем можно совершать, получая за них награду

Модель многорукого бандита

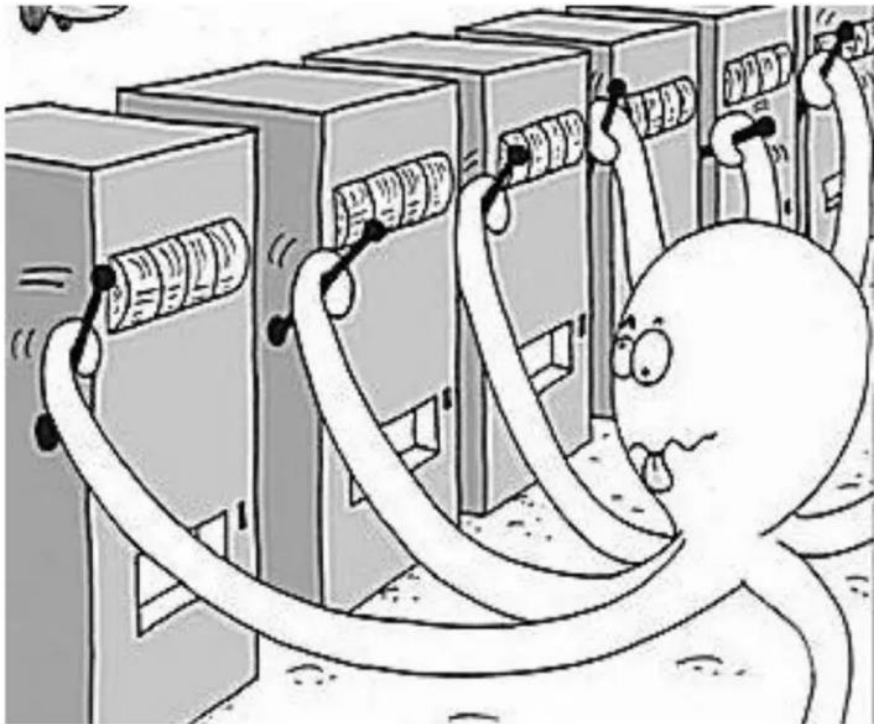


图1. MAB问题

- Постановка:
 - Есть k ручек, каждая из которых характеризуется распределением вознаграждения (неизвестным для агента)
 - На каждом раунде агент выбирает ручку и получает вознаграждение (в соответствии с распределением этой ручки)
- Цель:
 - Максимизировать суммарное (среднее) вознаграждение
- Конкретные приложения:
 - Реклама и рекомендации
 - Протоколы лечения

Многорукие бандиты. Пример 1

0: 90%, 20: 10%



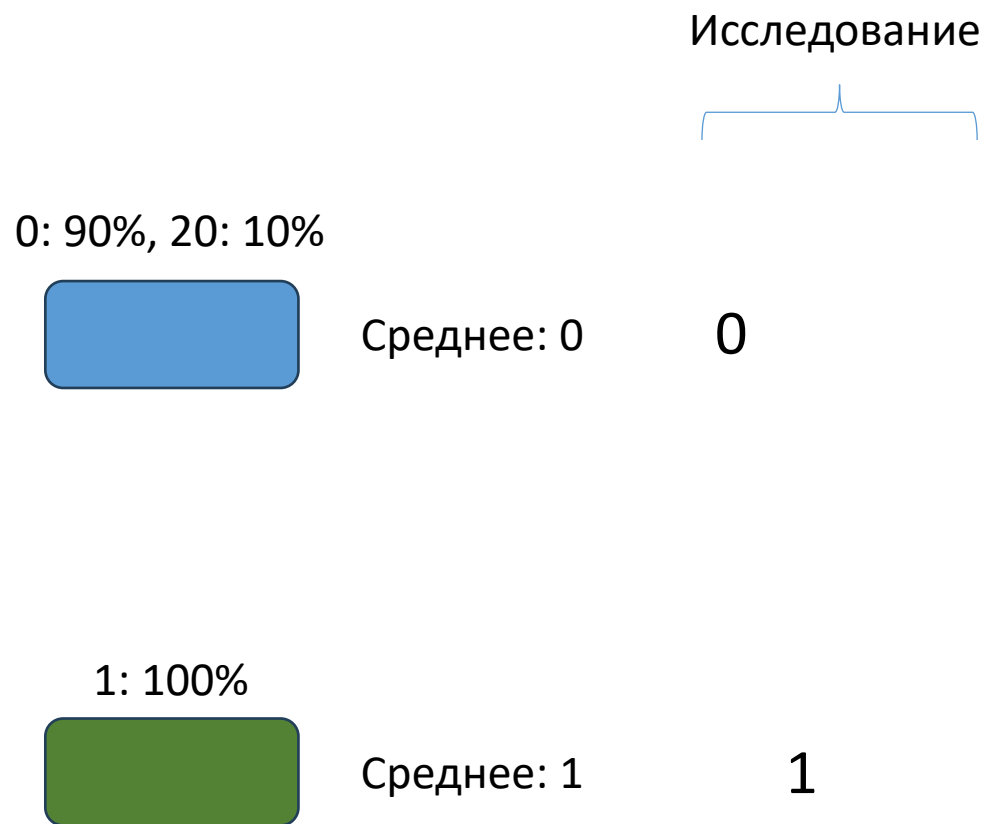
Среднее: 0

1: 100%

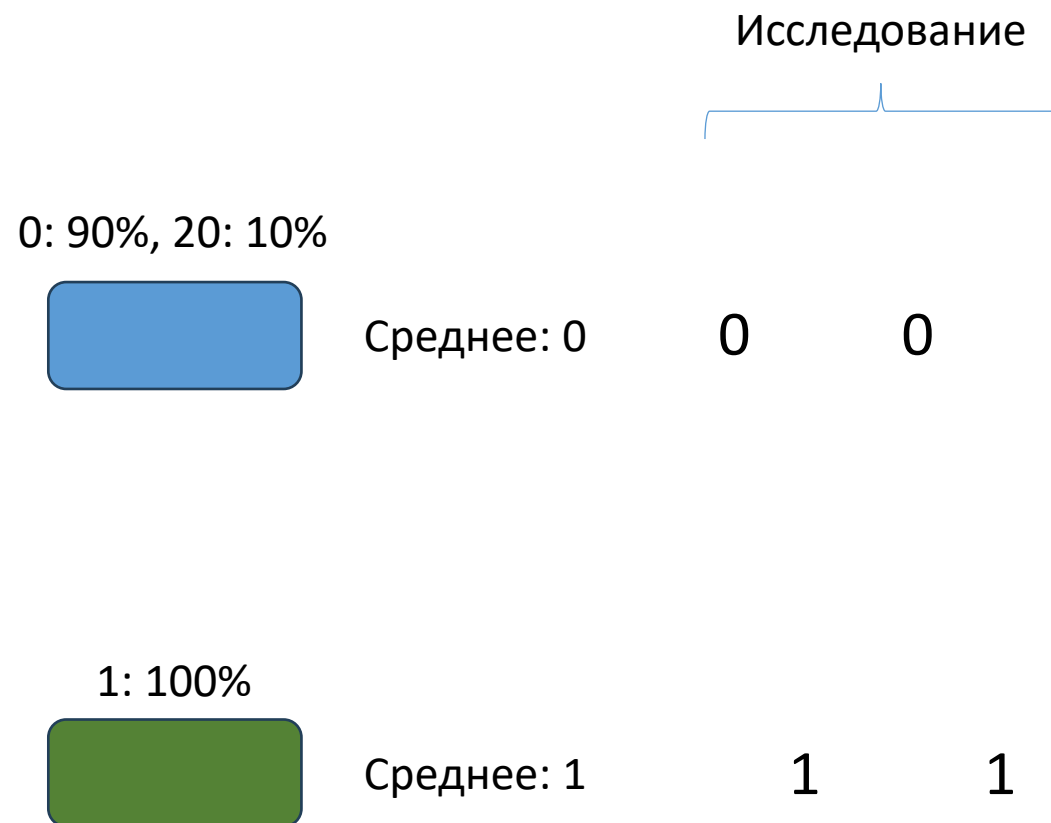


Среднее: 0

Многорукие бандиты. Пример 1



Многорукие бандиты. Пример 1



Многорукие бандиты. Пример 1



Так, конечно, вести себя **не стоит!**

Важнейшая задача: баланс между *исследованием* (получением новых данных) и *использованием* информации о наиболее выгодном действии (по имеющимся данным).

Многорукие бандиты. Исследование и использование

- Ценность действия a (неизвестная):

$$q_*(a) \stackrel{\text{def}}{=} \mathbb{E}[R_t | A_t = a]$$

- Оценка ценности действия a : $Q_t(a)$
- *Жадное* действие – действие, обладающее максимальной оценкой
 - Выбор жадного действия – *использование*
 - Другой выбор – *исследование*
- Если на шаге t выбираем *использование*, то не *исследуем* и наоборот => конфликт

Методы ценности действий

- Одна из (не единственная) оценка ценности действия:

$$Q_t(a) \stackrel{\text{def}}{=} \frac{\text{Сумма вознаграждений при выборе } a \text{ до момента } t}{\text{Сколько раз } a \text{ выбиралось до момента } t} =$$

$$= \frac{\sum_{i=1}^{t-1} R_i \mathbb{I}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i=a}}$$

- Если знаменатель равен 0, то $Q_t(a)$ принимается равным какому-либо числу (например, 0)
- При устремлении знаменателя к бесконечности $Q_t(a)$ по закону больших чисел сходится к $q_*(a)$

ε-жадный выбор действия

- Метод:

- Большую часть времени вести себя жадно:

$$A_t \stackrel{\text{def}}{=} \arg \max_a Q_t(a)$$

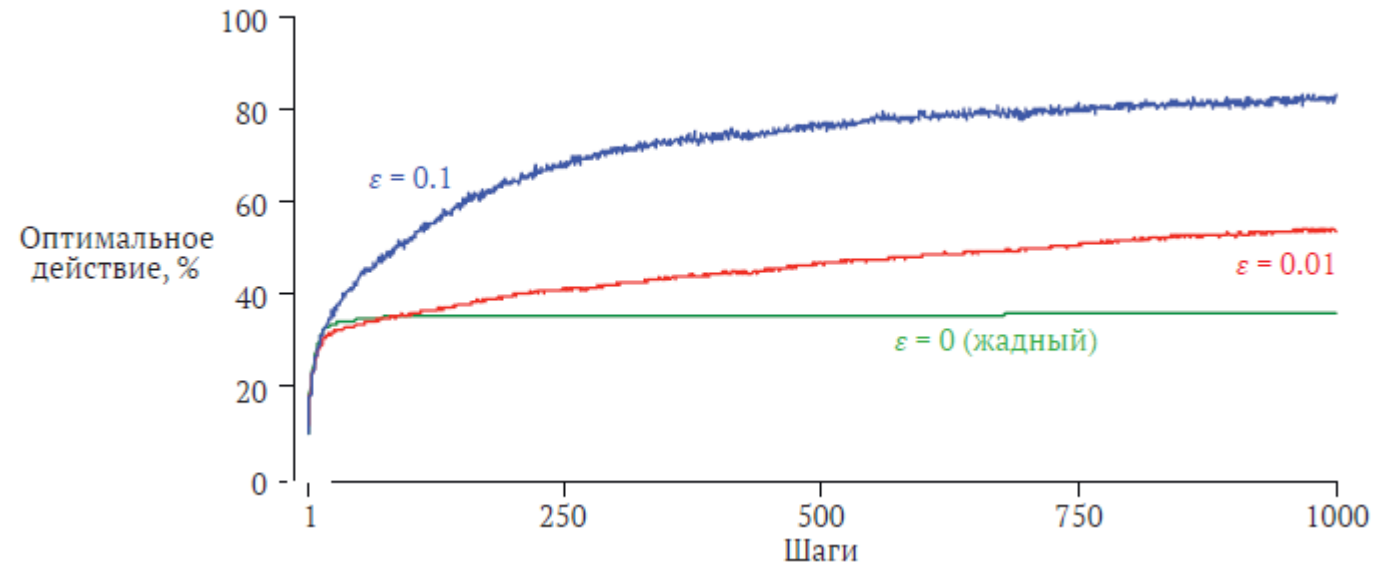
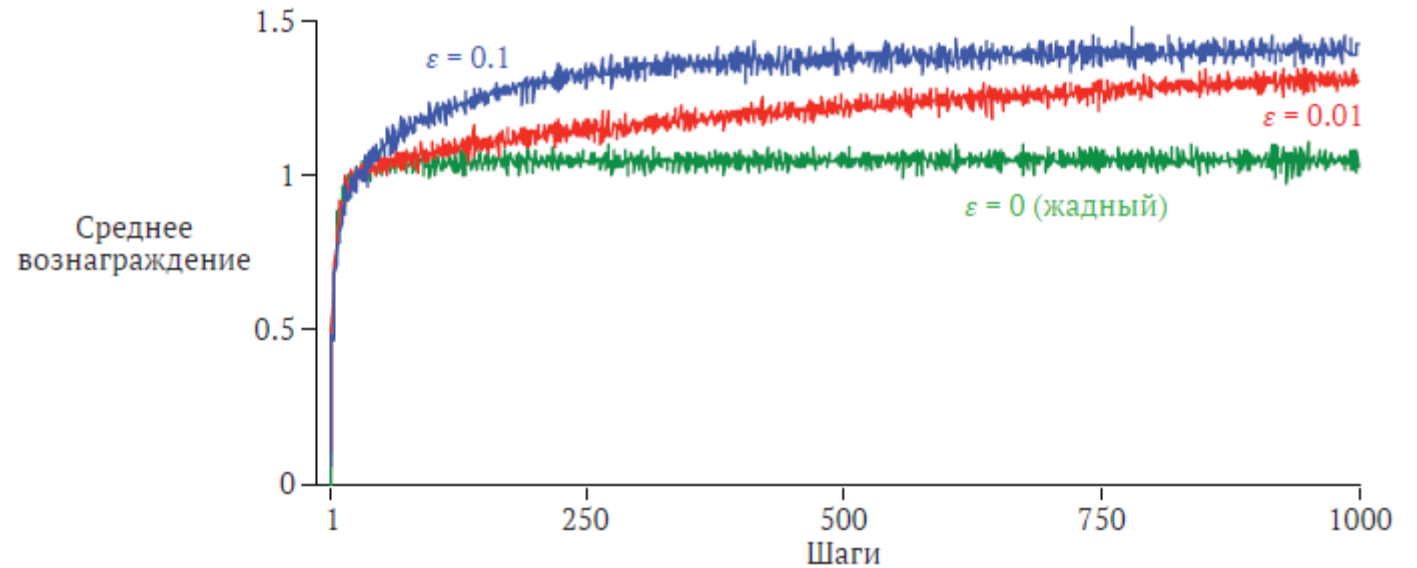
- Иногда, с малой вероятностью ε , выбирать действие случайным образом, не зависящим от оценок ценности действий

- Свойства:

- В *пределе*, при большом числе шагов, каждое действие будет выбрано *бесконечное* число раз, поэтому все $Q_t(a)$ сходятся к $q_*(a)$
 - Вероятность выбора оптимального действия сводится к числу, большему $1-\varepsilon$

ϵ -жадный выбор действия

- 10 «рук»
- Каждая формирует выигрыши в соответствии с нормальным распределением
- Результаты усреднены по нескольким прогонам
- Источник: Саттон и Барто



Инкрементная реализация

- Можно ли считать оценку ценности действия без хранения всей истории?
- Да:

$$\begin{aligned} Q_{n+1} &= \frac{1}{n} \sum_{i=1}^n R_i = \\ &= \frac{1}{n} \left(R_n + \sum_{i=1}^{n-1} R_i \right) = \frac{1}{n} \left(R_n + (n-1) \frac{1}{(n-1)} \sum_{i=1}^{n-1} R_i \right) = \\ &= \frac{1}{n} (R_n + (n-1)Q_n) = \frac{1}{n} (R_n + nQ_n - Q_n) = Q_n + \frac{1}{n} [R_n - Q_n] \end{aligned}$$

Инкрементная реализация

$$Q_{n+1} = Q_n + \frac{1}{n} [R_n - Q_n]$$

Разновидность более общей формы:

НоваяОценка = СтараяОценка + РазмерШага[Цель – СтараяОценка]

Простой алгоритм бандита

Простой алгоритм бандита

Инициализировать для a от 1 до k :

$$Q(a) \leftarrow 0$$

$$N(a) \leftarrow 0$$

Повторять бесконечно:

$$A \leftarrow \begin{cases} \operatorname{argmax}_a Q(a) & \text{с вероятностью } 1 - \varepsilon \\ \text{случайное действие} & \text{с вероятностью } \varepsilon \end{cases} \quad \begin{matrix} \text{(неоднозначность} \\ \text{разрешается} \\ \text{случайным образом)} \end{matrix}$$

$$R \leftarrow \text{bandit}(A)$$

$$N(A) \leftarrow N(A) + 1$$

$$Q(A) \leftarrow Q(A) + 1/N(A) [R - Q(A)]$$

Нестационарный вариант

- Особенность:
 - Свойства «ручек» меняются со временем

- Возможное решение:

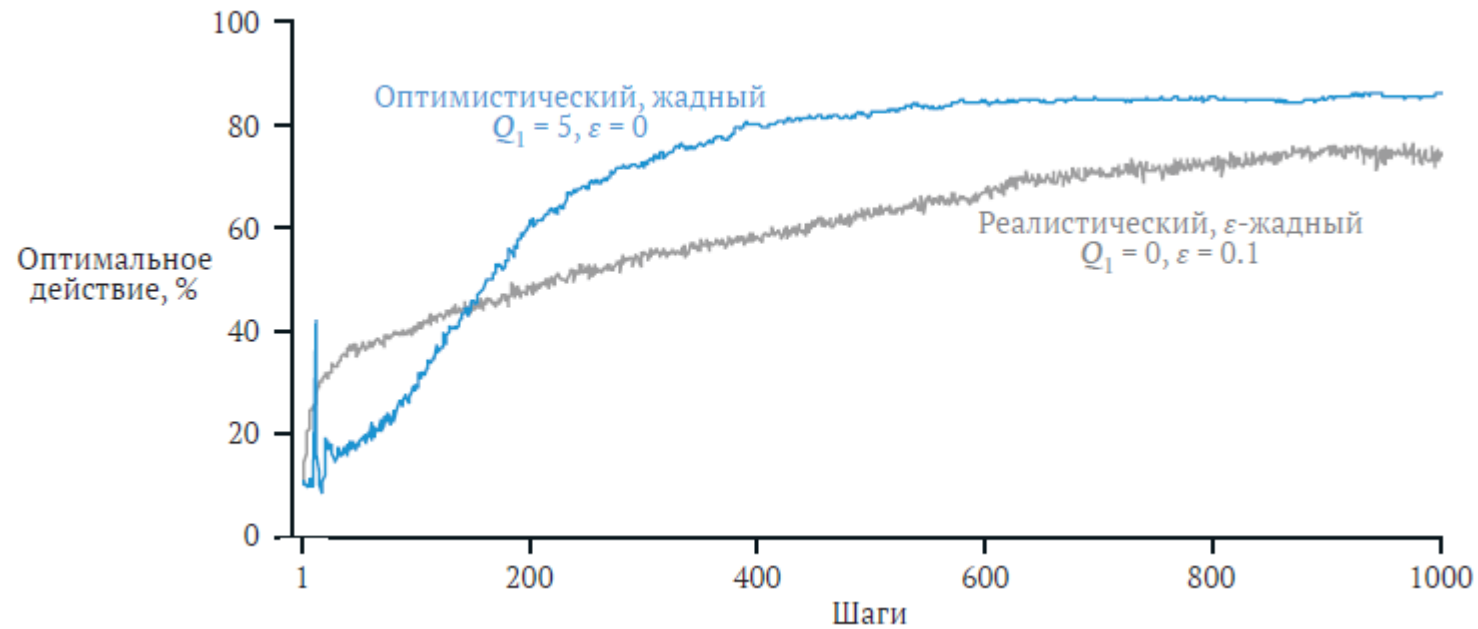
$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$

- Последствия:
 - $Q_{n+1} = (1 - \alpha)^n Q_1 + \sum_{i=1}^n \alpha(1 - \alpha)^{n-1} R_i$
 - (экспоненциально) взвешенное среднее, вес зависит от того, насколько давно встречалось то или иное R_i
 - экспоненциальное сглаживание
 - нарушается условие сходимости (но в нестационарном варианте это то, что нужно)

Оптимистические начальные значения

- Первый выбор зависит от $Q_1(a)$
- При использовании методов с выборочным средним это смещение исчезает после первого выбора каждого из действий, но не в случае с постоянным α !
- Можно использовать:
 - Задания априорных свидетельств о предпочтительности действий
 - Стимулирования исследования

Оптимистические начальные значения



UCB (Upper Confidence Bound).

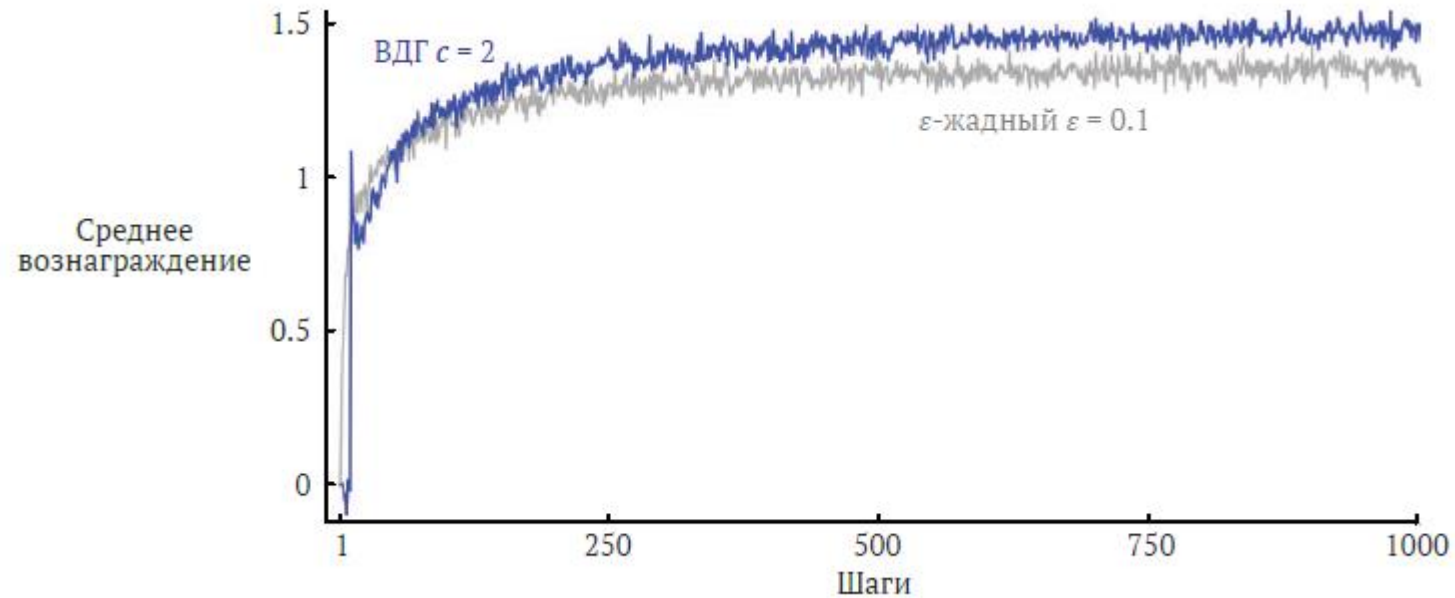
Верхняя доверительная граница (ВДГ)

- Идея:
 - Чем больше накоплено данных, тем наша оценка надежнее.
- Суть метода:
 - Выбор действия, в соответствии со следующим правилом:

$$A_t \stackrel{\text{def}}{=} \arg \max_a \left[Q_t(a) + c \sqrt{\frac{\ln t}{N_t(a)}} \right]$$

UCB (Upper Confidence Bound).

Верхняя доверительная граница (ВДГ)

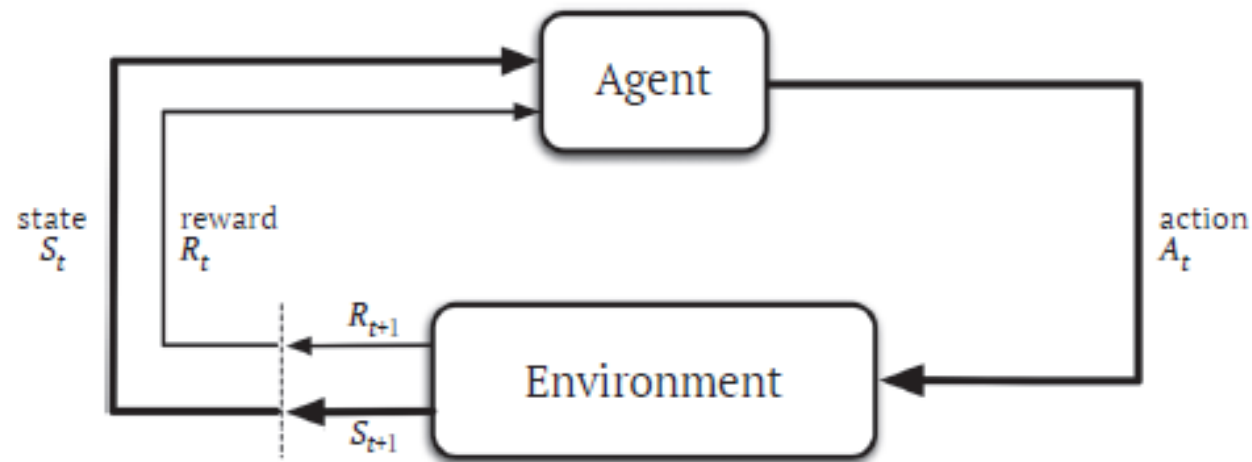


Тема 15. Обучение с подкреплением

Марковский процесс принятия решений (МППР)

- **Агент** – действующее лицо
- **Среда** – окружение, характеризуется состоянием (не все элементы которого могут быть наблюдаемы)
- **Действия** – агент может совершать действия, изменяющие состояние среды
- **Вознаграждение** – агент может получать вознаграждение за определенные действия
- **Политика/стратегия** (policy) – отображение наблюдаемых состояний в действия (что агент должен делать в той или иной ситуации)
- **Цель** – максимизация полного вознаграждения

Марковский процесс принятия решений (МППР)



- Взаимодействие происходит на каждом шаге *дискретной* последовательности шагов, $t = 0, 1, 2, \dots$
- На каждом шаге t агент:
 - Получает некоторое представление о состоянии среды $S_t \in \mathcal{S}$
 - Выбирает действие $A_t \in \mathcal{A}(s)$
- На следующем шаге агент, отчасти как следствие своего действия, получает числовое вознаграждение и оказывается в новом состоянии $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$
- Траектория: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \dots$
- Динамика МППР: $p(s', r | s, a) \stackrel{\text{def}}{=} \Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$

Марковский процесс принятия решений.

Примеры

- Биореактор:
 - Действия: целевая температура и скорость помешивания
 - Состояния: показания термопары и других датчиков, описание ингредиентов и конечного вещества
 - Вознаграждения: измерения скорости производства полезного химиката
- Робот-манипулятор:
 - Действия: напряжения, подаваемые на каждый мотор
 - Состояния: данные об углах поворота во всех сочленениях
 - Вознаграждение: +1 за успешно перемещенный объект, возможно, небольшое отрицательное число на каждом шаге

Доход и эпизоды

- Цель: максимизация ожидаемого дохода G_t

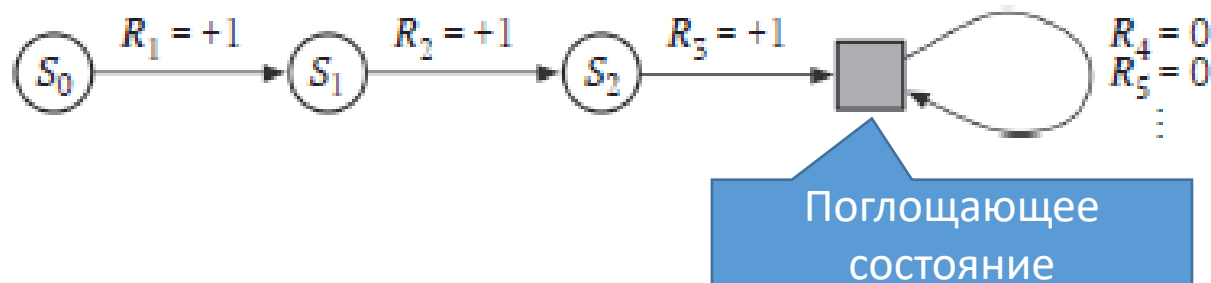
- Эпизодические задачи:

$$G_t \stackrel{\text{def}}{=} R_{t+1} + R_{t+2} + \dots + R_T$$

- Непрерывные задачи:

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

- Унификация (для удобства):



Стратегия и функция ценности

- **Стратегия** (π): отображение состояний на вероятности выбора для каждого возможного действия, $\pi(a|s)$
- **Функция ценности** $v_\pi(s)$ состояния s при стратегии π - ожидаемый доход, когда агент начинает работу в состоянии s и в дальнейшем следует стратегии π :

$$v_\pi(s) \stackrel{\text{def}}{=} E_\pi[G_t | S_t = s] = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s \right], \forall s \in \mathcal{S}$$

- **Функция ценности действий** $q_\pi(s, a)$ в состоянии s при стратегии π – ожидаемый доход, когда агент начинает работу в состоянии s , предпринимает действие a , а затем следует стратегии π :

$$q_\pi(s, a) \stackrel{\text{def}}{=} E_\pi[G_t | S_t = s, A_t = a] = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right]$$

Уравнения Беллмана

- Согласованность:

$$v_{\pi}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) [r + \gamma v_{\pi}(s')], \forall s \in \mathcal{S}$$

- Уравнения оптимальности:

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a) [r + \gamma v_*(s')]$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a) \left[r + \gamma \max_{a'} q_*(s',a') \right]$$

Прагматические задачи в контексте МППР

- Оценить стратегию
 - Понять, каков будет ожидаемый выигрыш, если мы начнем ей следовать из данного состояния
- Найти оптимальную стратегию

Алгоритм итеративного оценивания стратегии (Policy Evaluation)

Алгоритм итеративного оценивания стратегии для оценивания $V \approx v_\pi$

Вход: π , стратегия, подлежащая оцениванию

Параметр алгоритма: небольшая пороговая величина $\theta > 0$, определяющая точность оценки

Инициализировать $V(s)$ для всех $s \in \mathcal{S}^+$ произвольным образом с той оговоркой, что $V(\text{terminal}) = 0$

Повторять:

$\Delta \leftarrow 0$

 Повторять для каждого $s \in \mathcal{S}$:

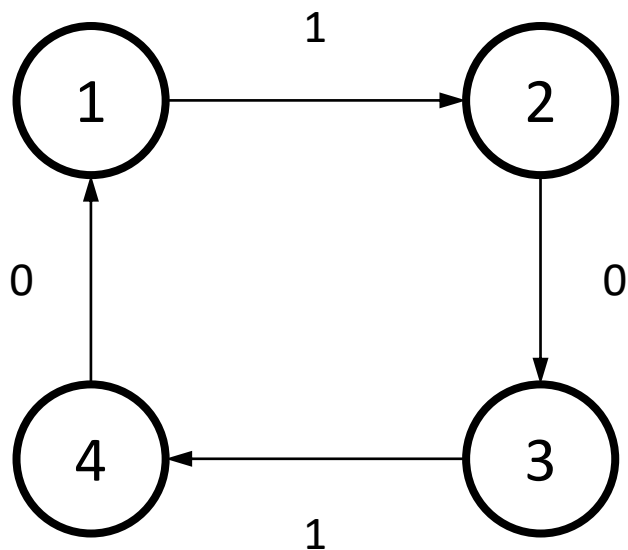
$v \leftarrow V(s)$

$V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) [r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

пока не окажется $\Delta < \theta$

Алгоритм итеративного оценивания стратегии (Policy Evaluation). Пример



$$\gamma = 0.5$$

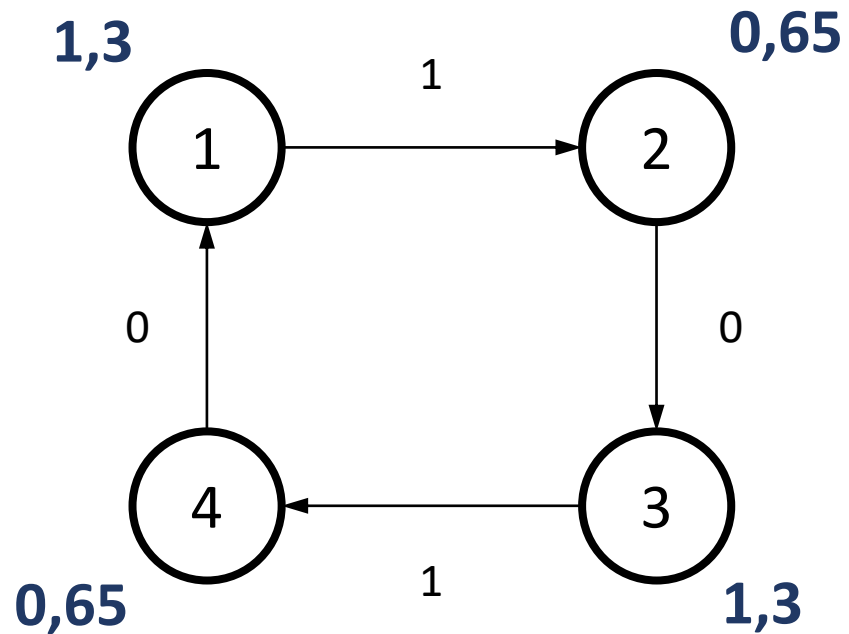
Алгоритм итерации по стратегиям (Policy Iteration)

Алгоритм итерации по стратегиям (с использованием итеративного оценивания стратегии) для оценивания $\pi \approx \pi_*$

1. Инициализация
 $V(s) \in \mathbb{R}$ и $\pi(s) \in \mathcal{A}(s)$ выбираются произвольно для всех $s \in \mathcal{S}$
2. Оценивание стратегии
Повторять:
 $\Delta \leftarrow 0$
 Повторять для каждого $s \in \mathcal{S}$:
 $v \leftarrow V(s)$
 $V(s) \leftarrow \sum_{s',r} p(s', r | s, \pi(s)) [r + \gamma V(s')]$
 $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
пока не окажется $\Delta < \theta$ (небольшое положительное число, определяющее точность оценки)
3. Улучшение стратегии
 $policy-stable \leftarrow true$
Для каждого $s \in \mathcal{S}$:
 $old-action \leftarrow \pi(s)$
 $\pi(s) \leftarrow \arg \max_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$
 Если $old-action \neq \pi(s)$, то $policy-stable \leftarrow false$
Если $policy-stable$, то остановиться и вернуть $V \approx v_*$ и $\pi \approx \pi'$; иначе перейти к 2

Алгоритм итерации по стратегиям (Policy Iteration). Пример

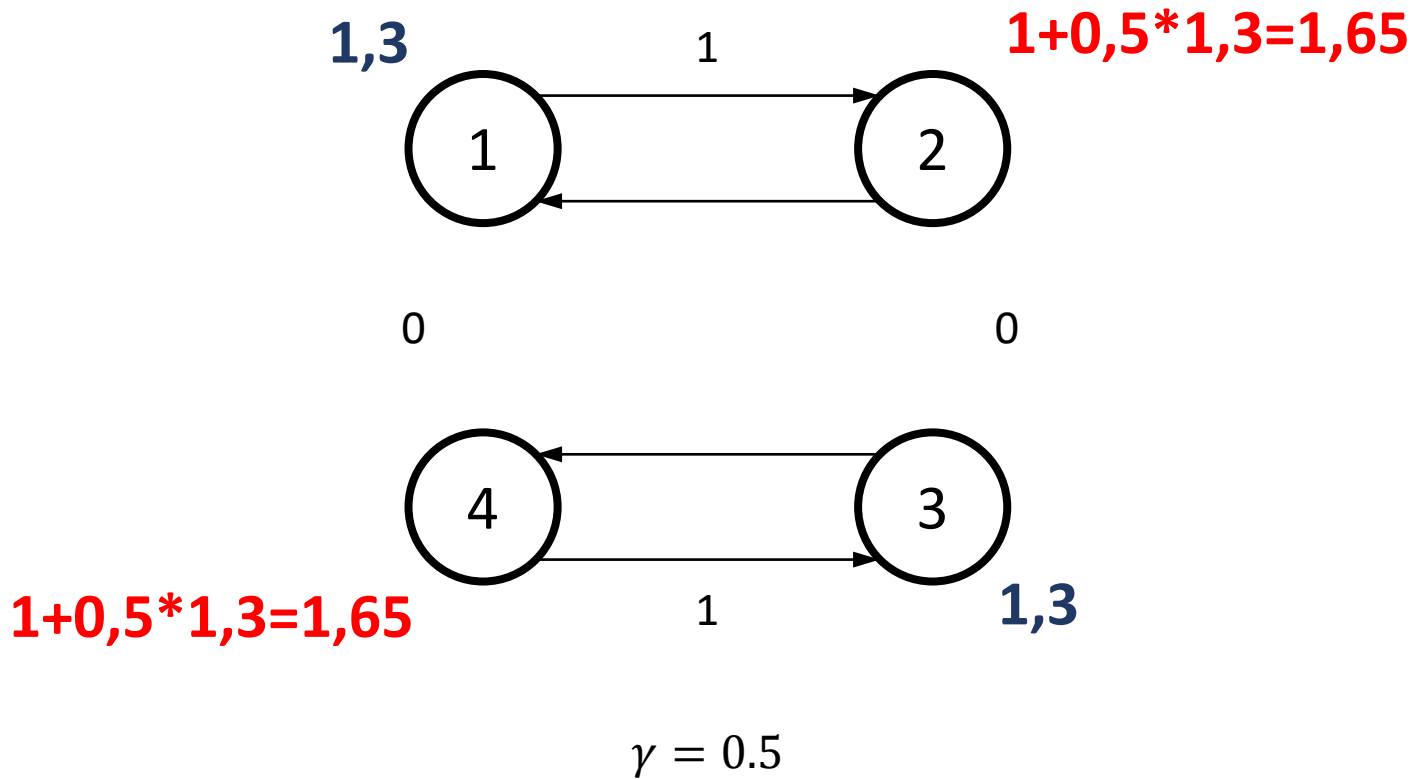
Оценили стратегию (п.2)



$$\gamma = 0.5$$

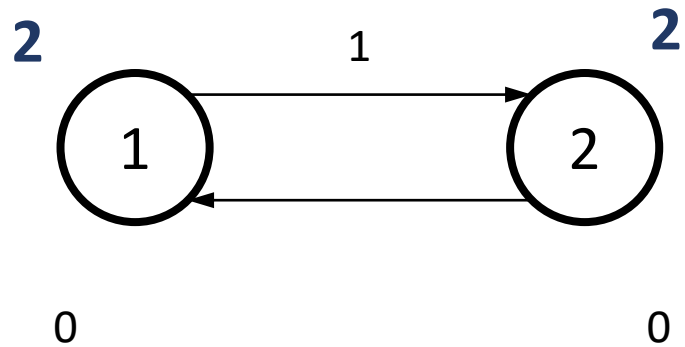
Алгоритм итерации по стратегиям (Policy Iteration). Пример

Нашли нестабильности и улучшили (п.3)



Алгоритм итерации по стратегиям (Policy Iteration). Пример

Оценили новую стратегию (п.2)



$$\gamma = 0.5$$

Недостатки и направление улучшения

- Основной недостаток:
 - На каждой итерации нужно оценивать всю стратегию, а это достаточно трудоемко
- Идея:
 - Алгоритм оценивания стратегии сходится лишь в пределе, но нужно ли дожидаться этой сходимости?

Алгоритм итерации по ценности (Value Iteration)

Алгоритм итерации по ценности для оценивания $\pi \approx \pi_*$.

Параметр алгоритма: небольшая пороговая величина $\theta > 0$, определяющая точность оценки

Инициализировать $V(s)$ для всех $s \in \mathcal{S}^+$ произвольным образом с той оговоркой, что $V(\text{terminal}) = 0$

Повторять:

$\Delta \leftarrow 0$

Повторять для каждого $s \in \mathcal{S}$:

$v \leftarrow V(s)$

$V(s) \leftarrow \max_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$

$\Delta \leftarrow \max(\Delta, |v - V(s)|)$

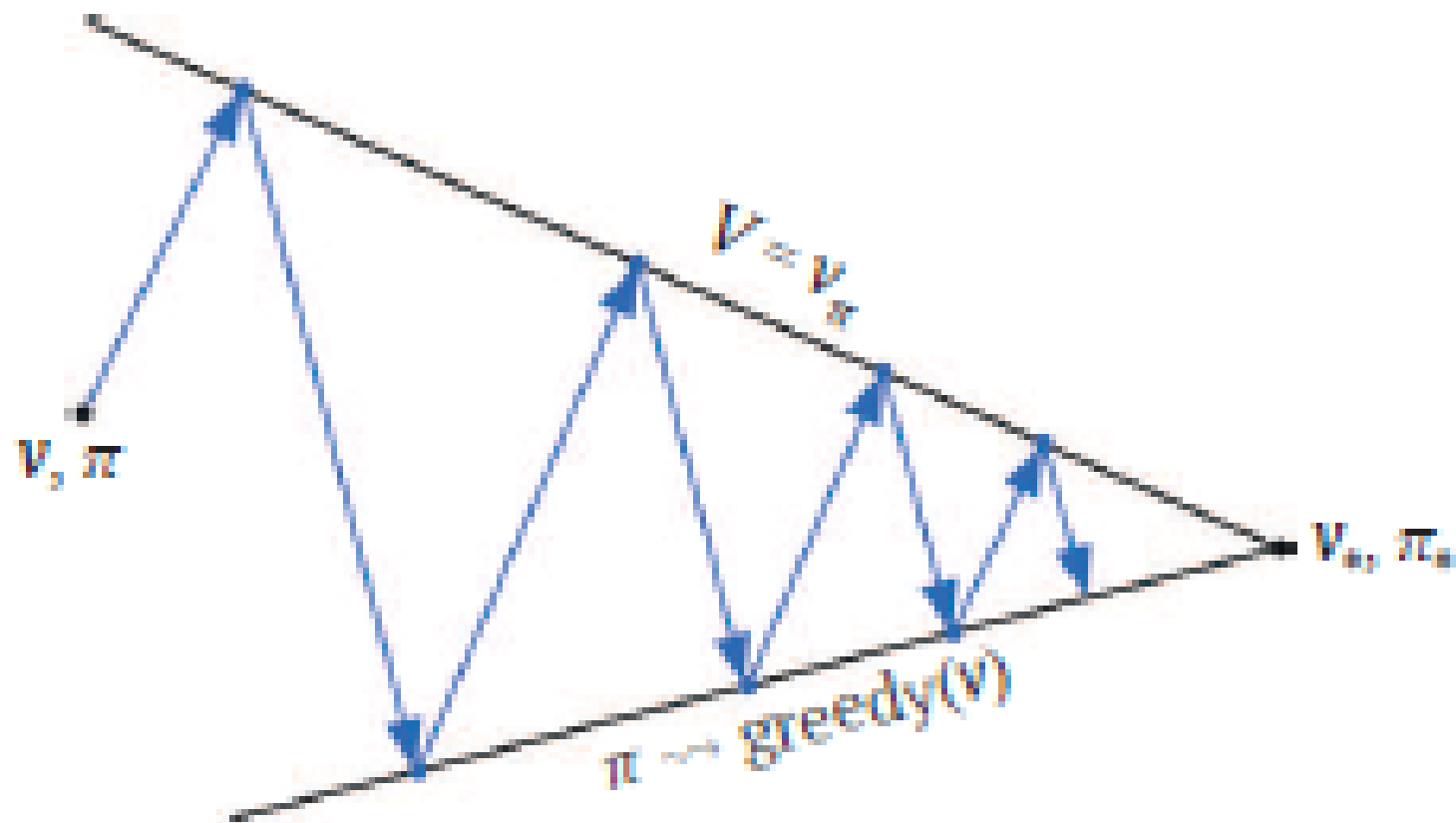
пока не окажется $\Delta < \theta$

Вывести детерминированную стратегию $\pi \approx \pi_*$ такую, что

$\pi(s) \leftarrow \arg \max_a \sum_{s',r} p(s', r | s, a) [r + \gamma V(s')]$

Объединение оценки и выбора
или
просто использование условия
оптимальности Беллмана

Обобщенная итерация по стратегиям (идея)



Промежуточное резюме

- Задача многоруких бандитов и дилемма между исследованием и эксплуатацией
 - ϵ -жадный
 - ВДГ
- МППР
 - Научились оценивать стратегию
 - Научились находить оптимальную стратегию
- Клиффхэнгер на следующую пару:
 - А что, если мы заранее не знаем динамики системы $p(s', r|s, a)$, т.е., ни модели переходов, ни выигрышей?

Задача обучения с подкреплением

- **Агент** – действующее лицо
- **Среда** – окружение, характеризуется состоянием (не все элементы которого могут быть наблюдаемы)
- **Действия** – агент может совершать действия, изменяющие состояние среды (*изначально он не знает модели изменения состояния*)
- **Вознаграждение** – агент может получать вознаграждение за определенные действия (*изначально он не знает модели назначения вознаграждений*)
- **Политика/стратегия** (policy) – отображение наблюдаемых состояний в действия (что агент должен делать в той или иной ситуации)
- **Цель** – максимизация вознаграждения

Задача обучения с подкреплением. Метафора

- **Агент** – действующее лицо
- **Среда** – окружение, характеризуется состоянием (не все элементы которого могут быть наблюдаемы)
- **Действия** – агент может совершать действия, изменяющие состояние среды (изначально он не знает модели изменения состояния)
- **Вознаграждение** – агент может получать вознаграждение за определенные действия (изначально он не знает модели назначения вознаграждений)
- **Политика/стратегия** (policy) – отображение наблюдаемых состояний в действия (что агент должен делать в той или иной ситуации)
- **Цель** – максимизация вознаграждения



Подходы в обучении с подкреплением

- Обучение с подкреплением с моделью (model-based)
 - Агент использует *модель изменения состояний* (для оценки возможных последствий действия и для оценки текущего состояния в условиях частично наблюдаемого состояния). Эта модель может быть неизвестна заранее, но строится по мере работы агента.
- Обучение с подкреплением без модели (model-free)
 - Агент не знает *модели изменения состояний* и не пытается ее построить, он пытается просто выучить политику действий.
 - Разновидности:
 - Оценка ценности (полезности) действий (action utility). Например, построение Q-функции, задающей полезность для заданного действия в заданном состоянии: $Q(s, a)$. Когда агент находится в определенном состоянии, он оценивает полезность всех доступных действий и выбирает действие, обладающее максимальной.
 - Поиск политики (policy search). Поиск непосредственно отображения из состояния в действие.

План

- ~~Методы, основанные на моделях~~
 - ~~Адаптивное динамическое программирование~~
- Model-free методы
 - TD(0)
 - SARSA
 - Q-Learning
- Аппроксимация функции ценности
 - Линейная аппроксимация (tile coding и пр.)
 - Нелинейная аппроксимация (нейронная сеть)
 - DQN
- Проблемы

Пассивное обучение с подкреплением

- Особенности:
 - Заранее неизвестна динамика МППР: $p(s', r|s, a)$
 - То есть, агент не знает последствий действий – переходов и вознаграждения
 - Задана фиксированная стратегия: $\pi(a|s)$
 - Цель:
 - Получить функции ценности $v_\pi(s)$, $q_\pi(s, a)$

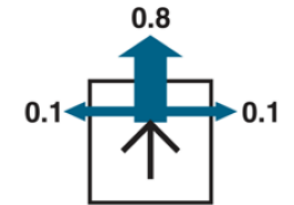
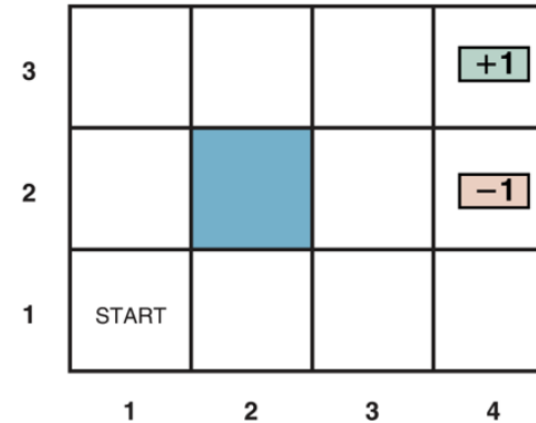
Пассивное обучение с подкреплением.

Простая идея

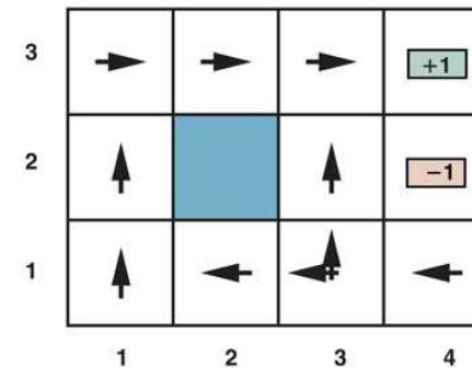
- Ценность состояния – это ожидаемое дисконтированное вознаграждение, которое можно получить, если, начав из этого состояния следовать стратегии
- Любая траектория представляет *одну выборку* этого значения для каждого посещенного состояния
- ***Значит***, нужно просто *следовать стратегии и накапливать* эти данные!

Пассивное обучение с подкреплением. Простая идея. Пример

$(1,1) \xrightarrow[-0.04]{Up} (1,2) \xrightarrow[-0.04]{Up} (1,3) \xrightarrow[-0.04]{Right} (1,2) \xrightarrow[-0.04]{Up} (1,3) \xrightarrow[-0.04]{Right} (2,3) \xrightarrow[-0.04]{Right} (3,3) \xrightarrow[-0.04]{Right} (4,3) \xrightarrow[-0.04]{+1}$
 $(1,1) \xrightarrow[-0.04]{Up} (1,2) \xrightarrow[-0.04]{Up} (1,3) \xrightarrow[-0.04]{Right} (2,3) \xrightarrow[-0.04]{Right} (3,3) \xrightarrow[-0.04]{Right} (3,2) \xrightarrow[-0.04]{Up} (3,3) \xrightarrow[-0.04]{Right} (4,3) \xrightarrow[-0.04]{+1}$
 $(1,1) \xrightarrow[-0.04]{Up} (1,2) \xrightarrow[-0.04]{Up} (1,3) \xrightarrow[-0.04]{Right} (2,3) \xrightarrow[-0.04]{Right} (3,3) \xrightarrow[-0.04]{Right} (3,2) \xrightarrow[-0.04]{Up} (4,2) \xrightarrow[-0.04]{-1}$



- Накопив такие данные, можно непосредственно оценить стоимости путей из состояния:
 - $(1, 1) \rightarrow (0.76 + 0.76 - 1.2) / 3 \sim 0.11$
 - ...



3	0.8516	0.9078	0.9578	<div>+1</div>
2	0.8016		0.7003	<div>-1</div>
1	0.7453	0.6953	0.6514	0.4279
	1	2	3	4

Пассивное обучение с подкреплением. Простая идея и ее существенный недостаток

Уравнение Беллмана, согласованность значений
ценности состояний:

$$v_{\pi}(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) [r + \gamma v_{\pi}(s')], \forall s \in \mathcal{S}$$



Am I a joke to you?

Пассивное обучение с подкреплением. Адаптивное динамическое программирование

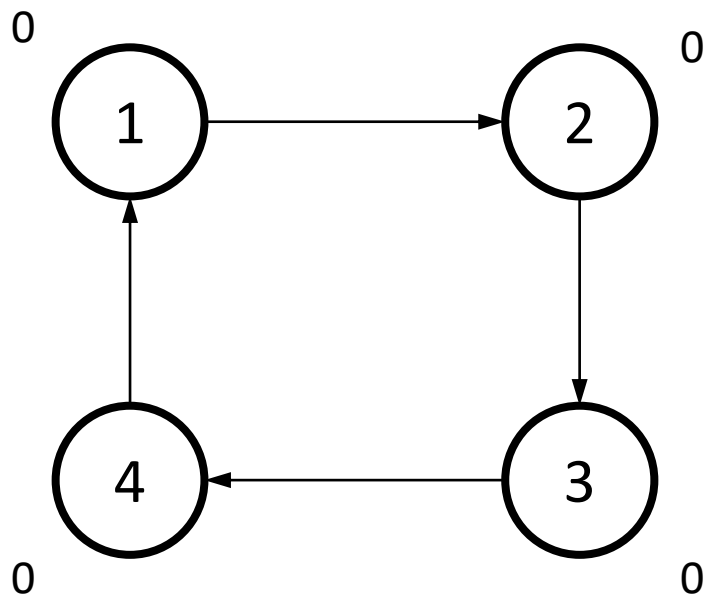
- Особенность:
 - Помимо сэмплирования, происходит и учет уравнений согласованности функции оценки состояний (уравнения Беллмана)
- Общая схема:
 - Сэмплируем, накапливаем траектории
 - Оцениваем функцию динамики марковского процесса $p(s', r|s, a)$
 - Она же модель переходов $p(s'|s, a)$
 - Она же – наблюдаемое вознаграждение $r(s, \pi(s), s')$
 - Используем эти оценки в алгоритме итеративного оценивания стратегии (Policy Evaluation)

Пассивное обучение с подкреплением. Адаптивное динамическое программирование

function PASSIVE-ADP-LEARNER(*percept*) **returns** an action
 inputs: *percept*, a percept indicating the current state s' and reward signal r
 persistent: π , a fixed policy
 mdp, an MDP with model P , rewards R , actions A , discount γ
 U, a table of utilities for states, initially empty
 $N_{s'|s,a}$, a table of outcome count vectors indexed by state and action, initially zero
 s, *a*, the previous state and action, initially null

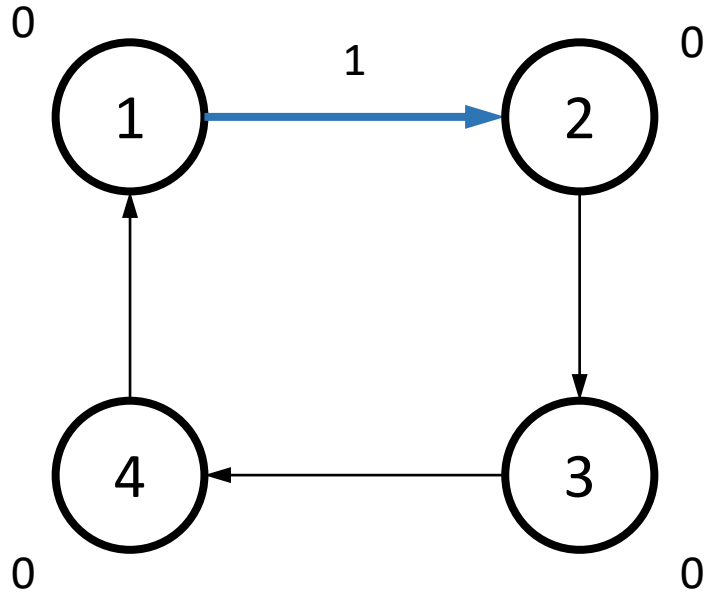
if s' is new **then** $U[s'] \leftarrow 0$
 if *s* is not null **then**
 increment $N_{s'|s,a}[s, a][s']$
 $R[s, a, s'] \leftarrow r$
 add *a* to $A[s]$
 $\mathbf{P}(\cdot \mid s, a) \leftarrow \text{NORMALIZE}(N_{s'|s,a}[s, a])$
 $U \leftarrow \text{POLICYEVALUATION}(\pi, U, \text{mdp})$
 $s, a \leftarrow s', \pi[s']$
 return *a*

Пассивное обучение с подкреплением.
TD (Temporal Difference). Временные различия



$$\gamma = 0.5$$

Пассивное обучение с подкреплением. TD (Temporal Difference). Временные различия



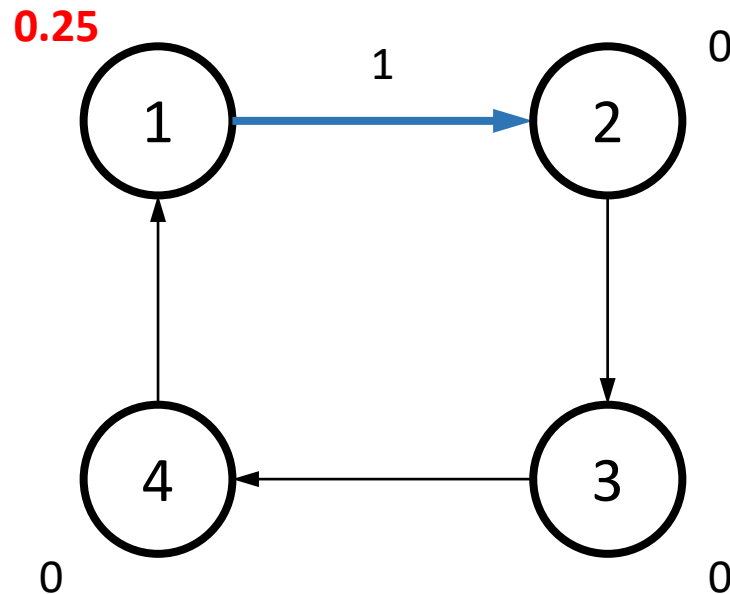
$$\gamma = 0.5$$

С другой стороны, с соответствии с уравнением Беллмана, мы ожидаем:

$$v_{\pi}(1) = 1 + \gamma v_{\pi}(2)$$

Пассивное обучение с подкреплением. TD (Temporal Difference). Временные различия

$$\alpha = 0.5$$



$$\gamma = 0.5$$

С другой стороны, с соответствии с уравнением Беллмана, мы ожидаем:

$$v_{\pi}(1) = 1 + \gamma v_{\pi}(2)$$

Скорректируем $v_{\pi}(1)$ в нужном направлении (с небольшим шагом α):

$$v_{\pi}(1) = v_{\pi}(1) + \alpha [\underbrace{1 + \gamma v_{\pi}(2)}_{\text{Цель, TD target}} - v_{\pi}(1)]$$

Цель,
TD target

Ошибка,
TD error

Пассивное обучение с подкреплением. TD (Temporal Difference). Временные различия

function PASSIVE-TD-LEARNER(*percept*) **returns** an action

inputs: *percept*, a percept indicating the current state s' and reward signal r

persistent: π , a fixed policy

s , the previous state, initially null

U , a table of utilities for states, initially empty

N_s , a table of frequencies for states, initially zero

if s' is new **then** $U[s'] \leftarrow 0$

if s is not null **then**

increment $N_s[s]$

$U[s] \leftarrow U[s] + \alpha(N_s[s]) \times (r + \gamma U[s'] - U[s])$

$s \leftarrow s'$

return $\pi[s']$

Активное обучение с подкреплением

- У агента нет фиксированной стратегии, он должен найти эту стратегию, взаимодействуя со средой (и, по возможности, оптимальную в смысле вознаграждения)
 - Для этого, в частности, нужно объединить два «мотива»:
 - Баланс между исследованием и использованием
 - Уравнения Беллмана

Активное обучение с подкреплением.

Адаптивное динамическое программирование

- Модель переходов можем обучать точно так же (ведя статистику последствий применения действий в том или ином состоянии)
- Не просто оцениваем заданную стратегию, а пытаемся получить оптимальную, для этого при пересчете используем *условие оптимальности* Беллмана:

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$$

- Выбор действия – тут нужно вспомнить про баланс между исследованием и использованием:
 - ϵ -жадно
 - случайно с вероятностью $1/t$ шаге
 - софтмакс
 - адаптация оптимистических значений или ДВГ

Активное обучение с подкреплением. Обучение без модели. Алгоритм Sarsa

Sarsa (TD-управление с единой стратегией) для оценивания $Q \approx q_*$

Параметры алгоритма: размер шага $\alpha \in (0, 1]$, небольшое $\varepsilon > 0$

Инициализировать $Q(s, a)$ для всех $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$ произвольным образом с ограничением $Q(\text{terminal}, \cdot) = 0$

Повторять для каждого эпизода:

 Инициализировать S

 Выбрать A в состоянии S , следуя стратегии, выведенной из Q (например, ε -жадной)

 Повторять для каждого шага эпизода:

 Предпринять действие A , наблюдать R, S'

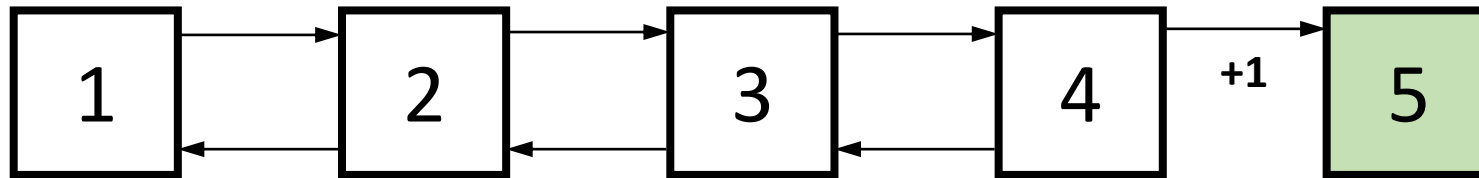
 Выбрать A' в S' , следуя стратегии, выведенной из Q (например, ε -жадной)

$Q(S, A) \leftarrow Q(S, A) + \alpha[R + Q(S', A') - Q(S, A)]$

$S \leftarrow S', A \leftarrow A'$;

 пока состояние S не является заключительным

Активное обучение с подкреплением. Обучение без модели. Алгоритм Sarsa



Активное обучение с подкреплением. Обучение без модели. Q-обучение

Q-обучение (TD-управление с разделенной стратегией) для оценивания $\pi \approx \pi_*$

Параметры алгоритма: размер шага $\alpha \in (0, 1]$, небольшое $\varepsilon > 0$

Инициализировать $Q(s, a)$ для всех $s \in \mathcal{S}^+$, $a \in \mathcal{A}(s)$ произвольным образом с ограничением $Q(\text{terminal}, \cdot) = 0$

Повторять для каждого эпизода:

 Инициализировать S

 Повторять для каждого шага эпизода:

 Выбрать A в состоянии S , следуя стратегии, выведенной из Q (например, ε -жадной)

 Предпринять действие A , наблюдать R, S'

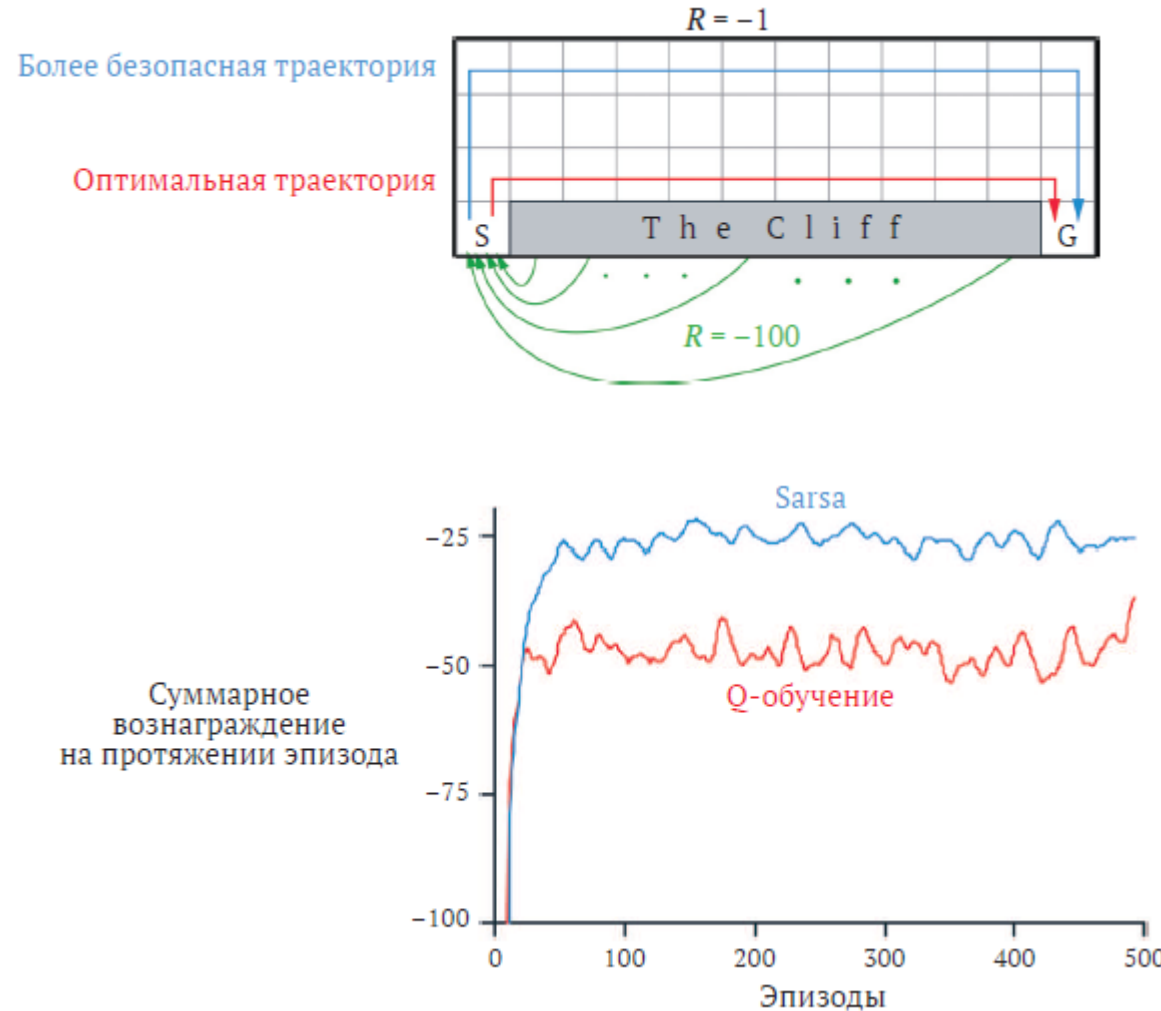
$Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$

$S \leftarrow S'$

пока состояние S не является заключительным

Активное обучение с подкреплением.

Единая и разделенная стратегии (on-policy и off-policy)



Аппроксимация функций ценности

- До этого момента рассматривался табличный вид функций $V(S)$ и $Q(S, A)$, иногда это может быть приемлемо, однако не в случае с непрерывным состоянием (или очень большим множеством состояний)
- Мы хотим, чтобы агент мог *обобщать* опыт для разных состояний
 - Для этого мы должны описывать состояния в виде набора признаков, и строить функции ценности, зависящие от сочетаний значений этих признаков

Аппроксимация функций ценности

- Градиентные методы:

$$w_{t+1} \stackrel{\text{def}}{=} w_t + \alpha[U_t - \hat{v}(S_t; w_t)]\nabla \hat{v}(S_t; w_t)$$

- Полуградиентные методы:

$$w_{t+1} \stackrel{\text{def}}{=} w_t + \alpha[R_t + \gamma \hat{v}(S_{t+1}; w_t) - \hat{v}(S_t; w_t)]\nabla \hat{v}(S_t; w_t)$$

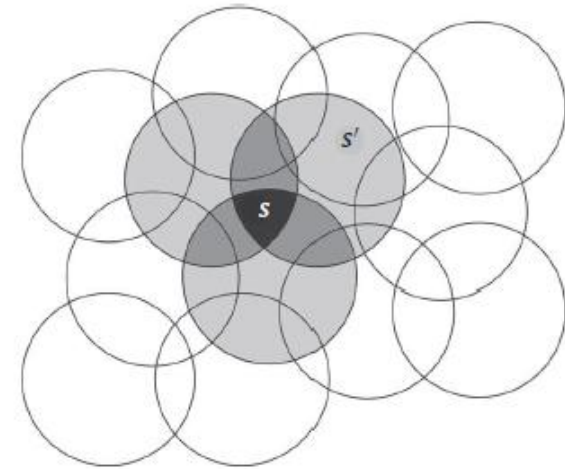
Линейная аппроксимация функций ценности

- Функция ценности:

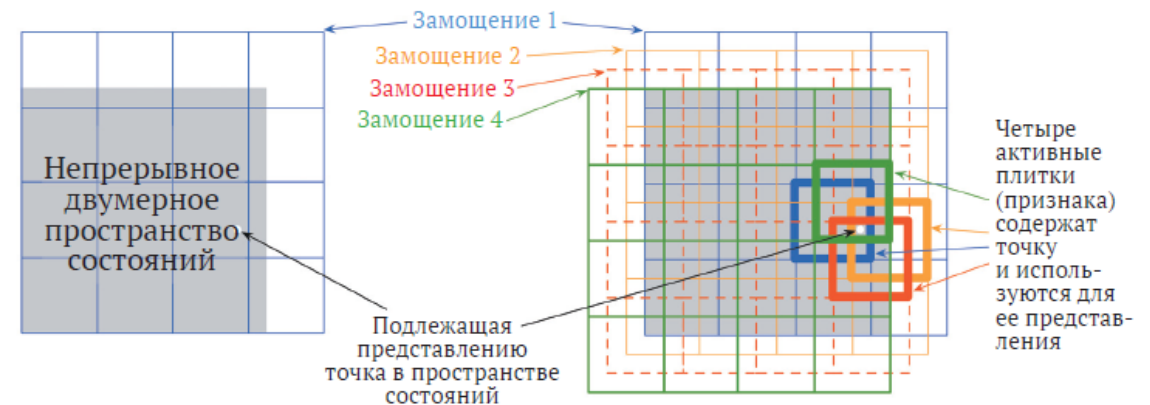
$$\hat{v}(s; w) \stackrel{\text{def}}{=} w^T x(s) = \sum_{i=1}^d w_i x_i(s)$$

- Правило корректировки весов:

$$w_{t+1} \stackrel{\text{def}}{=} w_t + \alpha[U_t - \hat{v}(S_t; w_t)]x(S_t)$$

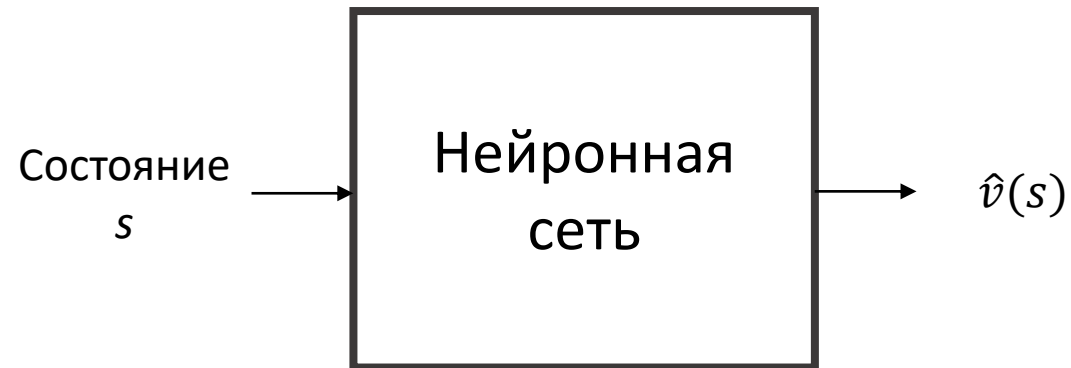
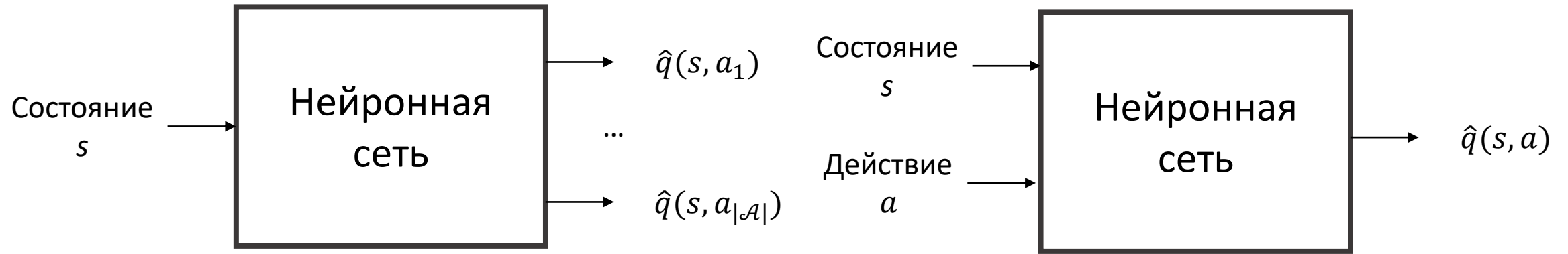


Грубое кодирование



Плиточное кодирование

Аппроксимация функций ценности с помощью нейронных сетей



Аппроксимация функций ценности с помощью нейронных сетей. Полугradientный Sarsa

Эпизодический полугradientный Sarsa для оценивания $\hat{q} = q_*$

Вход: дифференцируемая параметризация функции ценности действий \hat{q} :
 $\mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \rightarrow \mathbb{R}$

Параметры алгоритма: размер шага $\alpha > 0$, небольшое $\varepsilon > 0$

Инициализировать веса функции ценности $\mathbf{w} \in \mathbb{R}^d$ произвольным образом (например, $\mathbf{w} = \mathbf{0}$)

Повторять для каждого эпизода:

$S, A \leftarrow$ начальное состояние и действие эпизода (например, ε -жадное)

Повторять для каждого шага эпизода:

Предпринять действие A , наблюдать R, S'

Если S' заключительное:

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha[R - \hat{q}(S, A, \mathbf{w})]\nabla \hat{q}(S, A, \mathbf{w})$$

Перейти к следующему эпизоду

Выбрать A' как функцию от $\hat{q}(S, \cdot, \mathbf{w})$ (например, ε -жадное)

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha[R + \gamma \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})]\nabla \hat{q}(S, A, \mathbf{w})$$

$S \leftarrow S'$;

$A \leftarrow A'$;

Обучение с подкреплением. Итоги

- Что нужно:
 - Достаточно хорошая модель системы (имитатор) или возможность экспериментов в реальном мире
- Опасности:
 - Смертельная триада (риск неустойчивости и расходимости):
 - аппроксимация функций (например, линейная аппроксимация или ИНС)
 - Бутстрэппинг (обновление, основанное на оценках, а не фактическом вознаграждении и полном доходе)
 - обучение с разделенной стратегией (off-policy)

Краткое резюме

1. При обучении стратегии действий важен баланс между использованием и исследованием
 - ϵ -жадность и др.
2. Пассивное обучение с подкреплением (нет модели, известна стратегия)
 - Адаптивное динамическое программирование («блужданием» обучаем модель переходов и ее же используем для уточнения оценок состояний и действий)
 - На основе временных различий (приближаемся к аппроксимации уравнений Беллмана при каждом переходе)
3. Активное обучение с подкреплением (нет ни модели, ни стратегии)
 - Sarsa
 - Q-обучение
4. Аппроксимация функций ценности (для обобщения)
 - Линейная аппроксимация
 - Нелинейная (нейронные сети)

Литература

- Основное:

- Саттон Р. С., Барто Э. Дж. Обучение с подкреплением: Введение. 2-е изд. / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2020. – 552 с.: ил.
- Рассел С., Норвиг П. Искусственный интеллект. Современный подход. 4-е изд.

- Дополнительно:

- Лекции D. Silver: <https://www.youtube.com/watch?v=2pWv7GOvuf0>
- Специализация на Coursera (от университета Ватерлоо, где работают Саттон и Барто): <https://www.coursera.org/specializations/reinforcement-learning>

Последняя лекция

1. Построение интеллектуальных агентов, на основе знаний о проблеме (требуется моделирование человеческих знаний):
 1. Поиск (описание задачи, модель переходов)
 2. Логические агенты (в частности, онтологии, позволяющие рассуждать о терминологии проблемной области)
 3. Вероятностные модели (байесовские сети)
 4. Нечеткие модели
2. Построение самообучающихся агентов, использующих примеры и опыт:
 1. Машинное обучение как минимизация эмпирической ошибки, нейронные сети как достаточно универсальная абстракция, пригодная для параметрического обучения
 2. Компьютерное зрение
 3. Обработка текстов на естественном языке
 4. Обучение с подкреплением – обучение *действовать*