

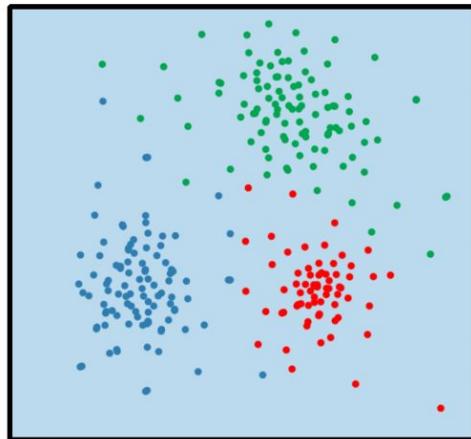
# Reinforcement Learning

обучение с подкреплением

Денисов Степан  
Милько Андрей  
Добросовестнов Иван

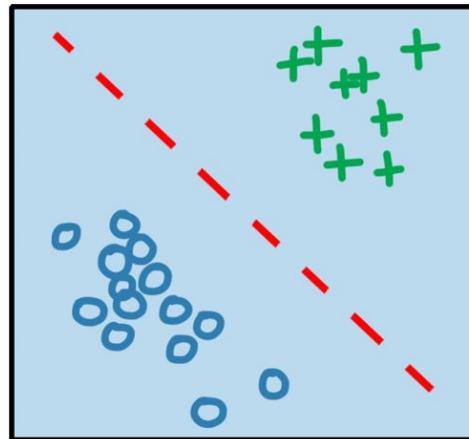
# machine learning

unsupervised  
learning



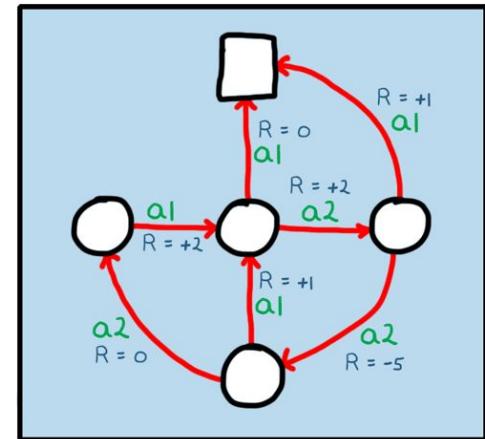
data driven

supervised  
learning



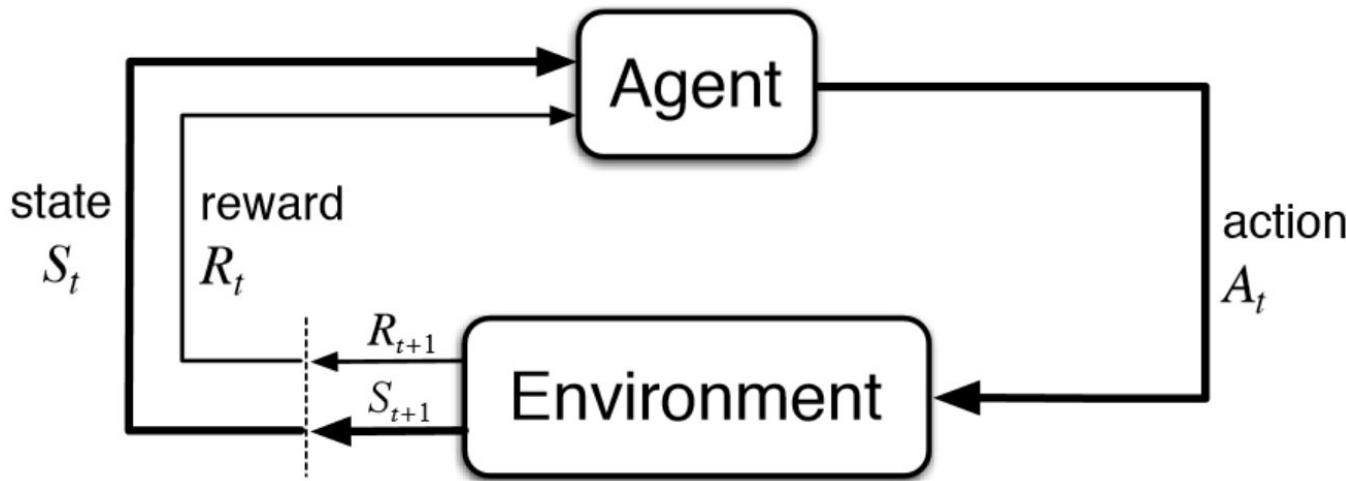
task driven

reinforcement  
learning

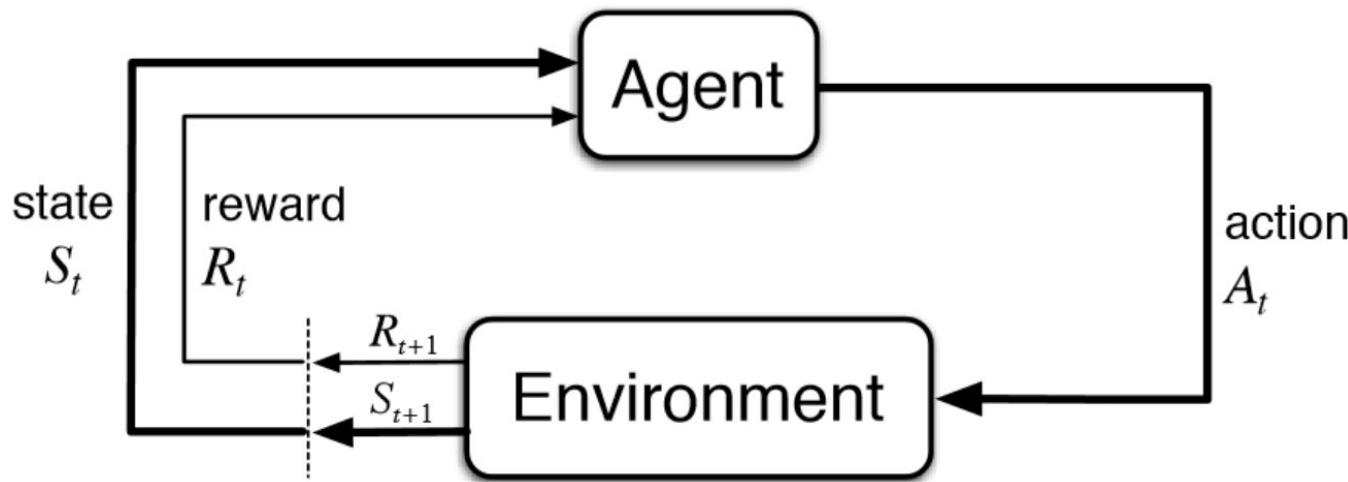


learn from mistakes

# Общая постановка задачи



# Общая постановка задачи

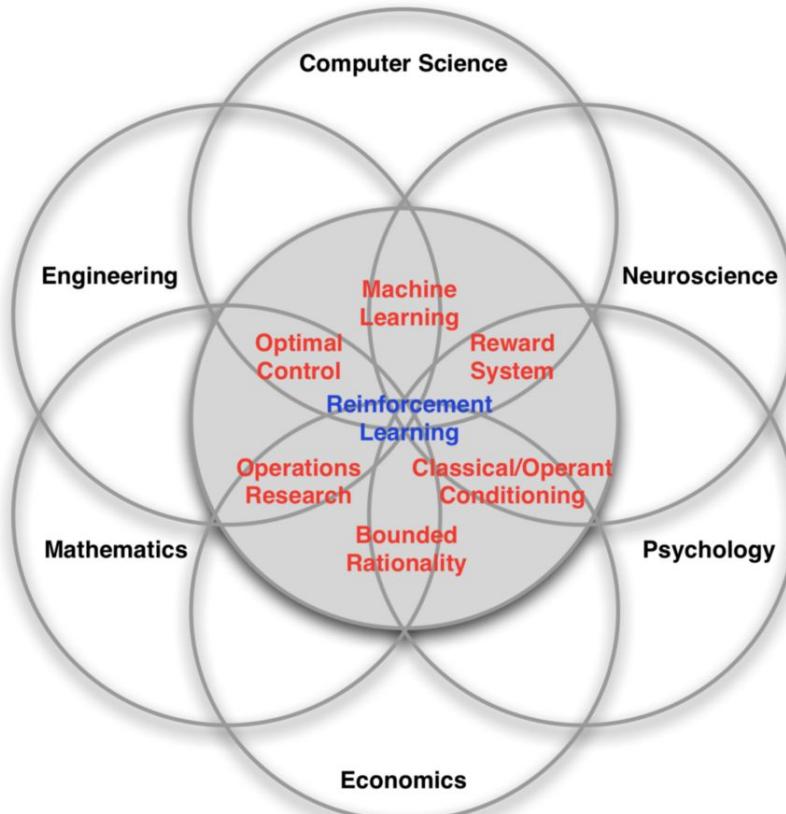


**Обучение интеллектуального агента  
хорошой последовательности принятия решений  
в условиях неполной информации  
в целях максимизации вознаграждения**

# Почему RL != unsupervised learning?

<b>Unsupervised Learning</b>	<b>Reinforcement Learning</b>
есть функция потерь	ошибка задается косвенно, через reward
обучение на основе данных	обучение оптимальной стратегии путем проб и ошибок
не нужен фидбэк	необходим фидбэк по действиям агента
модель не влияет на входные данные	агент может влиять на свои собственные наблюдения
последовательность данных не имеет значения	время имеет особое значение – данные последовательны

# Области применения

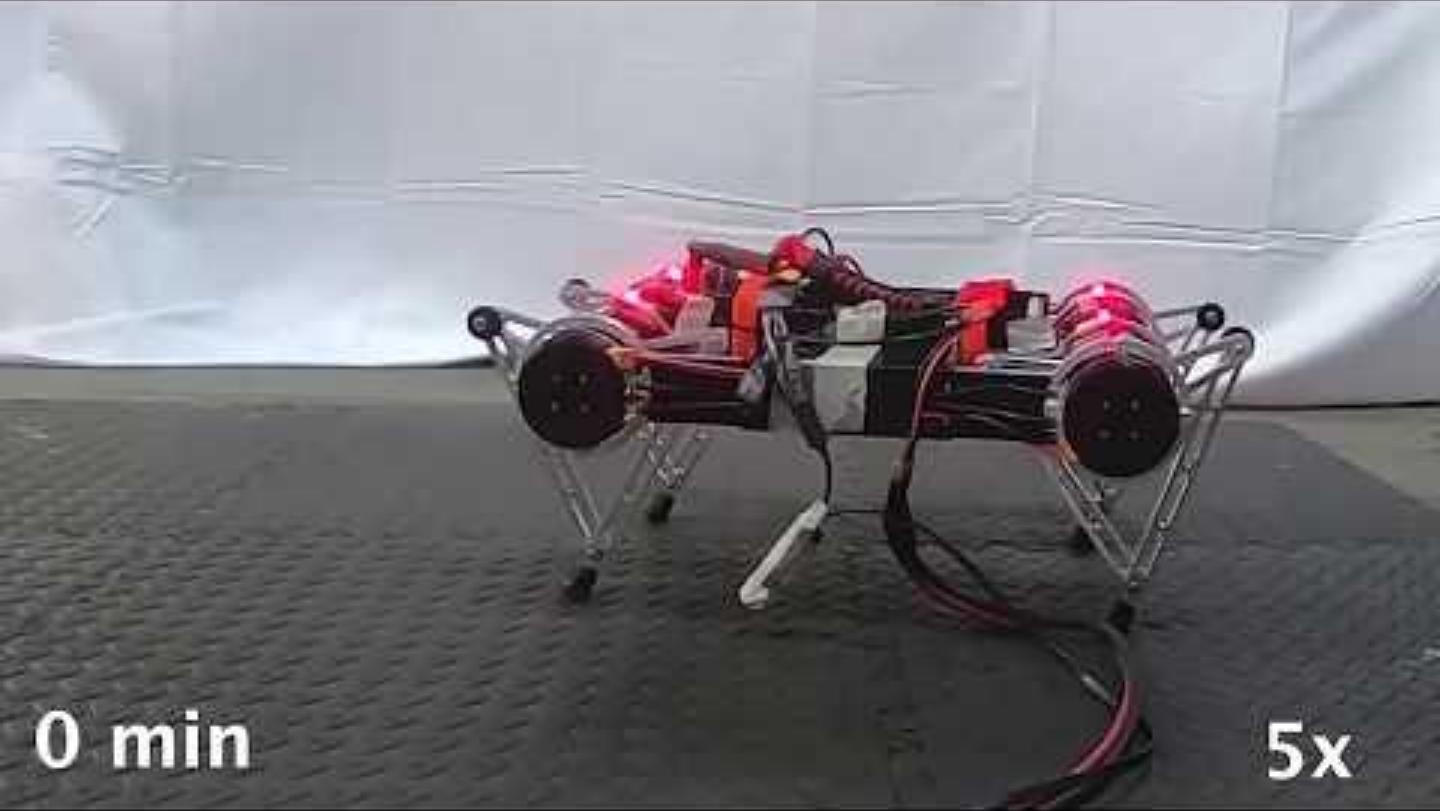


# **Case-study**

**Дано:**  
робот с ногами

**Цель:**  
научиться ходить вперед





0 min

5x



**Hurricane**

## **2 основные проблемы**

1. Не всегда очевидно, что в действительности означает “оптимальное действие”
2. Если всегда следовать текущей оптимальной стратегии, то можно никогда не достичь чего-то лучшего

# **Строение RL агента**

- Стратегия (policy function) – функция поведения агента
- Функция полезности (value function) – оценка того, насколько полезно состояние
- Модель (model) – представление агента о среде

# Policy function

- Детерминированная стратегия:

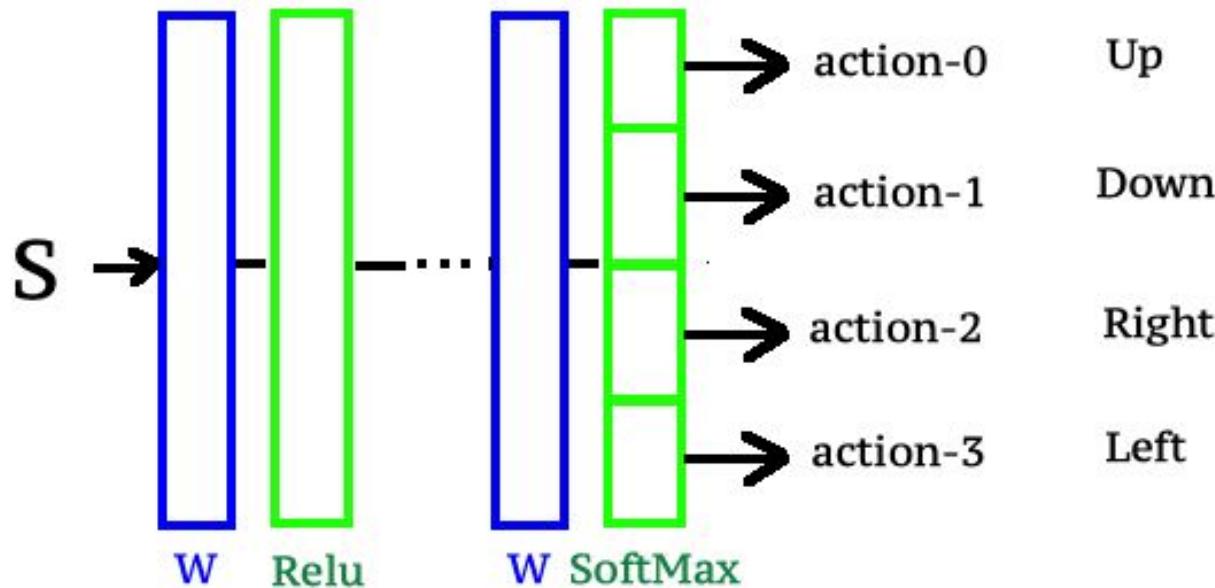
$$a = \pi(s)$$

- Стохастическая стратегия:

$$\pi(a|s) = P[a_t = a | s_t = s]$$

# Обучение policy function

$$\pi(a|s)$$



# **Value function**

Функция полезности – это предсказание будущего суммарного вознаграждения из состояния  $s$ .

$$V^\pi = \mathbb{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots | s_t = s]$$

Используется для оценки состояний.

# Model

Модель служит для предсказания того, что произойдет в среде в следующий момент времени.

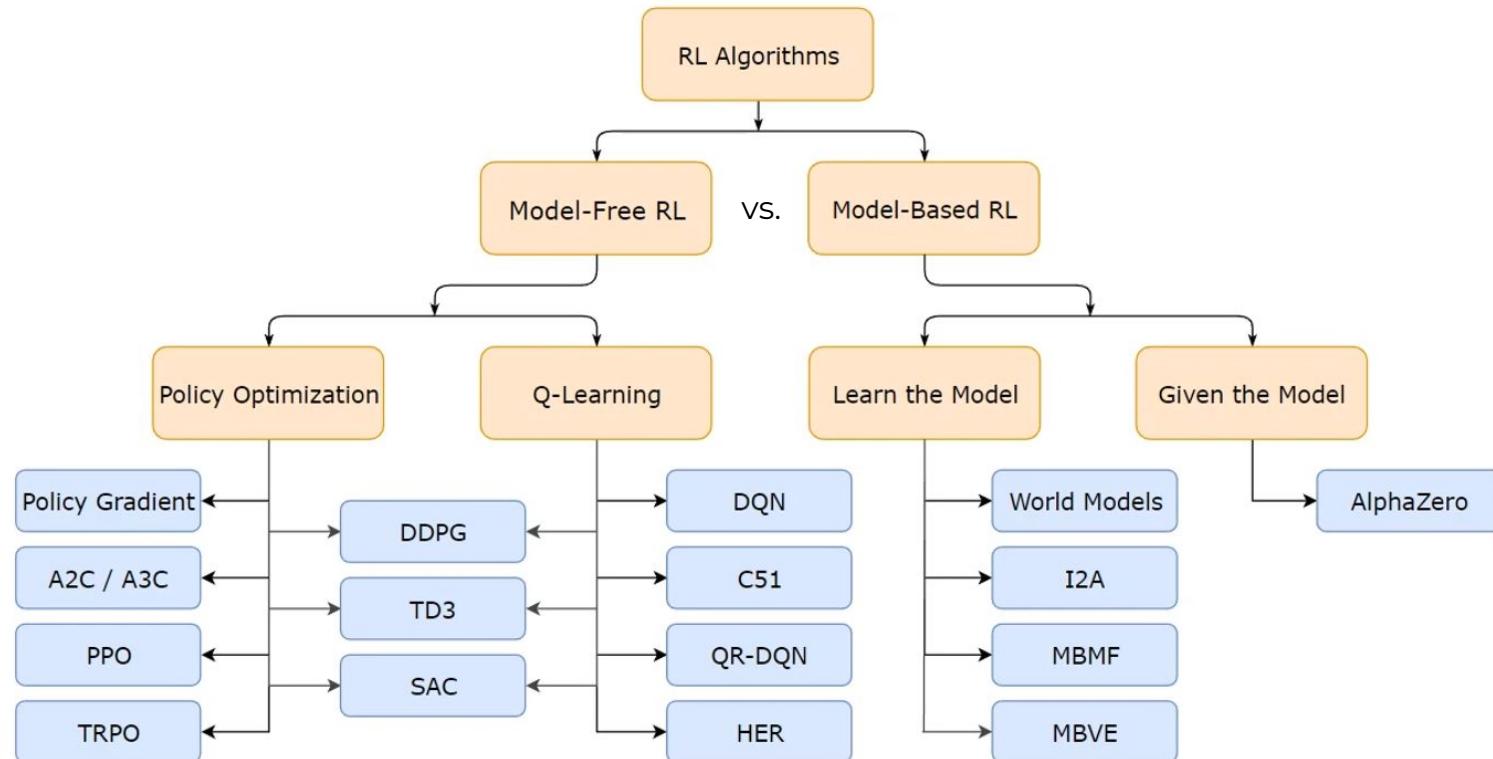
- Модель переходов  $P$  предсказывает следующее состояние:

$$\mathcal{P}_{ss'}^a = \mathbb{P}[s_{t+1} = s' | s_t = s, a_t = a]$$

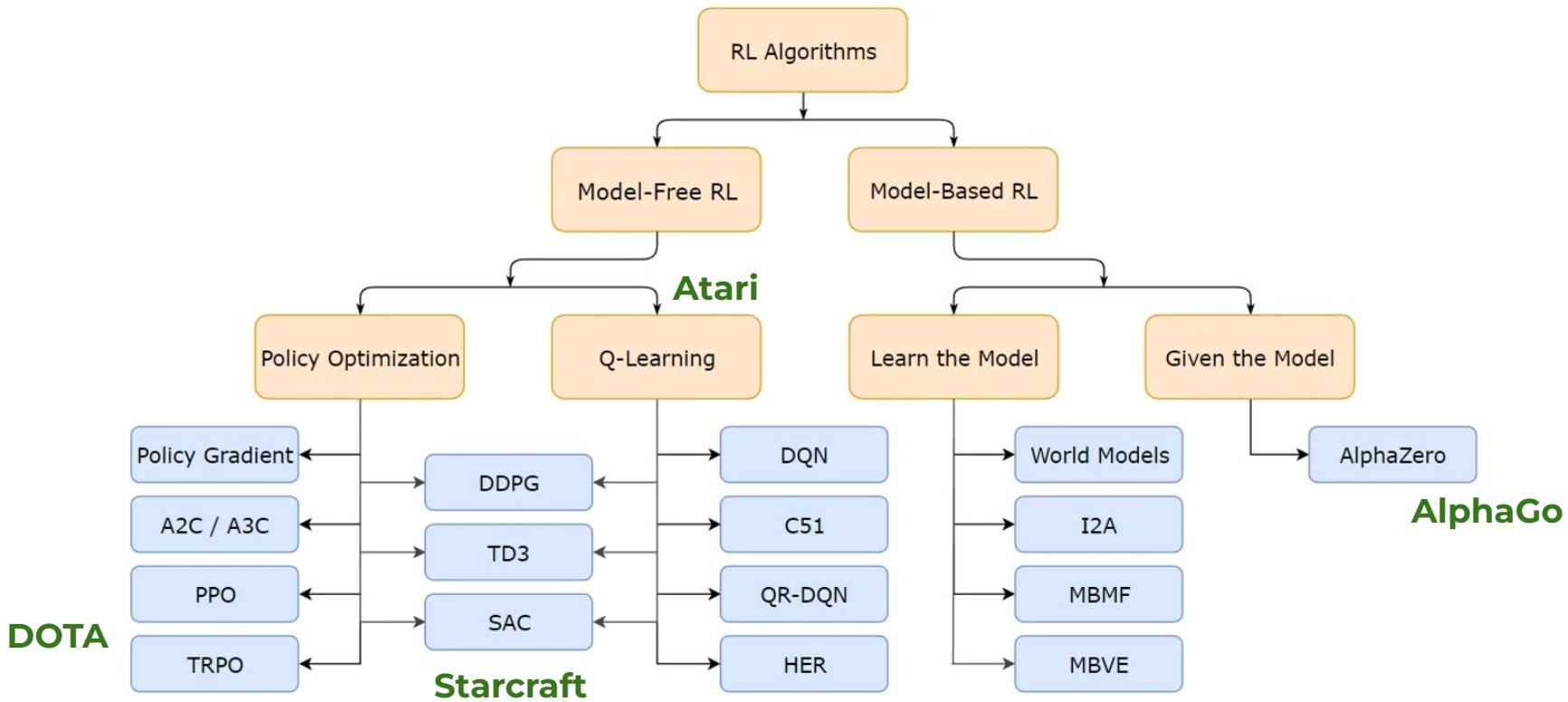
- Модель вознаграждений  $R$  предсказывает следующий reward:

$$\mathcal{R}_s^a = \mathbb{P}[r_t | s_t = s, a_t = a]$$

# Вариативность методов



# Вариативность методов



# Типизация RL агентов

- Оценивающие функцию полезности (value-based)
- Оценивающие стратегию (policy-based)
- Оценивающие обе функции (actor-critic)

# Markov Assumption

Опр. Состояние  $s$  называется **марковским**,  
если оно содержит всю необходимую информацию,  
которую нужно знать для принятия решения

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}) = P(s_{t+1}|s_t, a_t)$$

# **Markov Decision Process**

MDP – математический объект, описывающий принцип взаимодействия агента со средой:

$(S, A, P, R, \gamma)$ , где:

- $S$  – конечное множество состояний
- $A$  – конечное множество действий
- $P$  – модель переходов состояний
- $R$  – модель вознаграждений
- $\gamma$  – множитель дисконтирования

Отдельно выделяют Partially Observable MDP (POMDP), если агент наблюдает состояние не полностью

# Reinforcement learning is easy!

1. take action



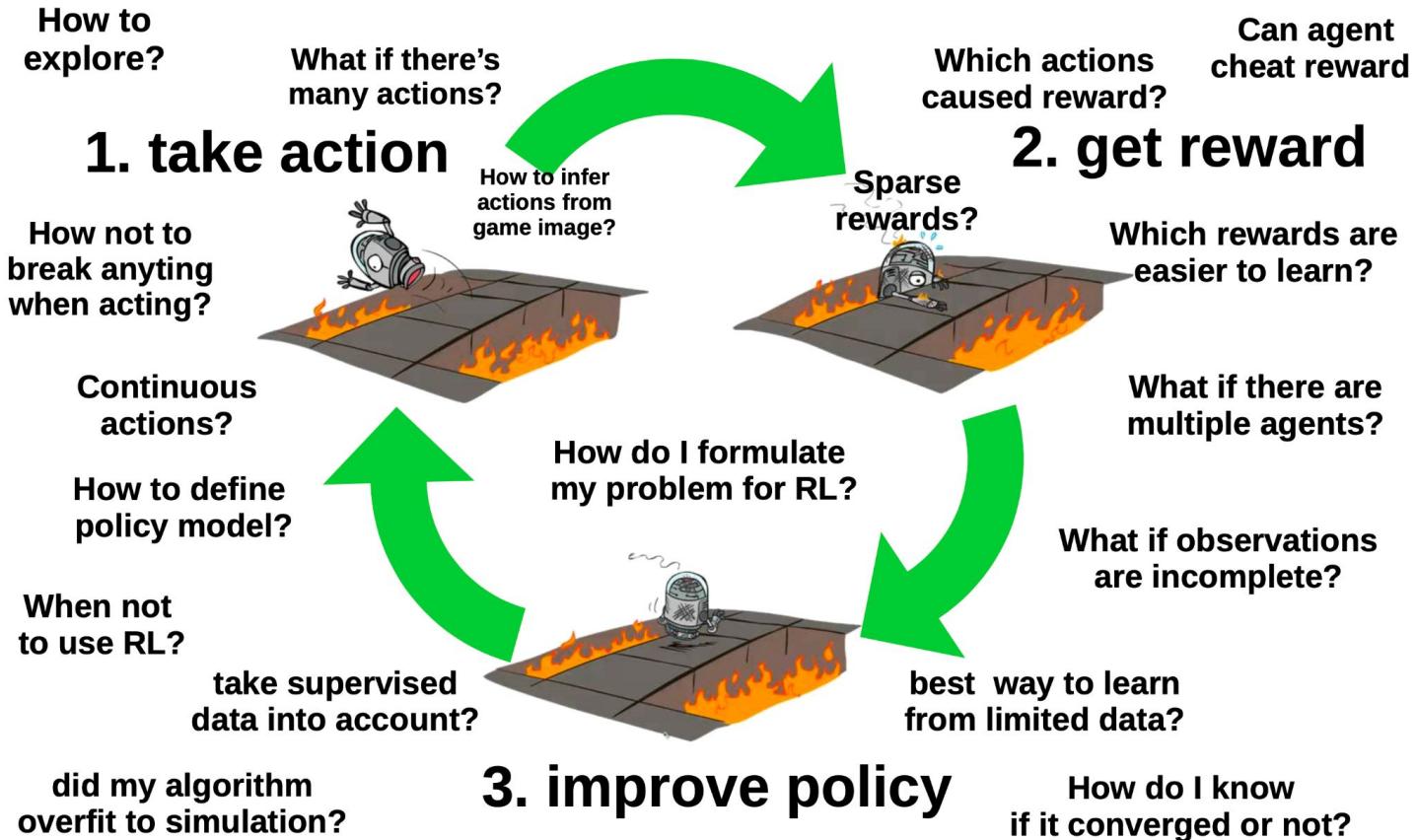
2. get reward



3. improve policy



# Reinforcement learning is challenging!





I KNOW

REINFORCEMENT LEARNING

# **Markov Decision Process (MDP)**

Процесс принятия решений по Маркову

- Уравнение Беллмана
- Q-функция

# **Уравнение Беллмана**

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии  
(Policy Functions) и оптимальные функции ценности (Value Functions)

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии  
(Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут  
разные функции ценности в соответствии с разными стратегиями

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии  
(Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии  
(Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

Уравнение Беллмана утверждает, что функция ценности  
(Value Function) может быть разложена на две части

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии (Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

Формула:  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

Уравнение Беллмана утверждает, что функция ценности (Value Function) может быть разложена на две части

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии (Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

Уравнение Беллмана утверждает, что функция ценности (Value Function) может быть разложена на две части

- **вознаграждение при переходе в следующее состояние**

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии (Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$

Уравнение Беллмана утверждает, что функция ценности (Value Function) может быть разложена на две части

- вознаграждение при переходе в следующее состояние

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии (Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

Уравнение Беллмана утверждает, что функция ценности (Value Function) может быть разложена на две части

- **вознаграждение при переходе в следующее состояние**
- **ценность за нахождение в следующем состоянии**

# Уравнение Беллмана

Помогает находить оптимальные функции стратегии (Policy Functions) и оптимальные функции ценности (Value Functions)

Мы знаем, что наша стратегия постоянно меняется, поэтому у нас будут разные функции ценности в соответствии с разными стратегиями

Формула:  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$

Уравнение Беллмана утверждает, что функция ценности (Value Function) может быть разложена на две части

- вознаграждение при переходе в следующее состояние
- ценность за нахождение в следующем состоянии

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

$s' = S_{t+1}$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $\underline{s'}$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E} [R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, **насколько хорошо было для робота находиться в этом состоянии  $s$**

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

Используя уравнение Беллмана, мы можем сказать, что это равно ожиданию от вознаграждения, которое робот получил при выходе из состояния  $s$ , и ценности нового состояния  $s'$ , куда переехал робот

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

Используя уравнение Беллмана, мы можем сказать, что это равно ожиданию от вознаграждения, которое робот получил при выходе из состояния  $s$ , и ценности нового состояния  $s'$ , куда переехал робот

# Уравнение Беллмана

Давайте поймем, о чём говорит эта формула с помощью примера

**Формула:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$

## Пример для понимания

Предположим, что робот находится в каком-то состоянии  $s$ , затем он переходит из этого состояния в какое-то другое состояние  $s'$ . Теперь вопрос в том, насколько хорошо было для робота находиться в этом состоянии  $s$

Используя уравнение Беллмана, мы можем сказать, что это равно ожиданию от вознаграждения, которое робот получил при выходе из состояния  $s$ , и ценности нового состояния  $s'$ , куда переехал робот

# **Q-функция**

Эта функция определяет, насколько хорошо для агента предпринимать действие **a** в состоянии **s** с политикой **π**

# Q-функция

Эта функция определяет, насколько хорошо для агента предпринимать действие **a** в состоянии **s** с политикой **π**

**Формула:**  $q_\pi(s, a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right].$

# Q-функция

Эта функция определяет, насколько хорошо для агента предпринимать действие **a** в состоянии **s** с политикой **π**

**Формула:**  $q_\pi(s, a) \doteq \mathbb{E}_\pi[G_t | S_t = s, A_t = a] = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right].$

# Q-функция

Эта функция определяет, насколько хорошо для агента предпринимать действие **a** в состоянии **s** с политикой **π**

**Формула:**  $q_{\pi}(s, a) \doteq \mathbb{E}_{\pi}[G_t \mid S_t = s, A_t = a] = \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right].$

returns - общая сумма вознаграждения агента за каждое его последующее состояние

# Q-функция

Эта функция определяет, насколько хорошо для агента предпринимать действие **a** в состоянии **s** с политикой **π**

**Формула:**  $q_\pi(s, a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right].$

По сути, эта формула говорит нам о ценности выполнения определенного действия **a** в состоянии **s** с политикой **π**

# **Markov Decision Process (MDP)**

Процесс принятия решений по Маркову

- **Уравнение ожидания Беллмана**
  - **для Value Function**
  - **для Q-функции**

# Ожидание Беллмана

**Уравнение Беллмана:**

$$v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

# Ожидание Беллмана

**Уравнение Беллмана:**

$$v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

**Уравнение ожидания Беллмана:**

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

# Ожидание Беллмана

**Уравнение Беллмана:**

$$v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

**Уравнение ожидания Беллмана:**

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

Разница между ур-ием Беллмана и ур-ием ожидания Беллмана в том, что во 2-м случае мы находим ценность конкретного состояния  $s$ , подвергнутого некоторой политике  $\pi$ , то есть из изменений в формуле здесь только введение политики  $\pi$

# Ожидание Беллмана

**Уравнение Беллмана:**  $v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$

**Уравнение ожидания Беллмана:**  $v_\pi(s) = \mathbb{E}_\pi [R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$

Разница между ур-ием Беллмана и ур-ием ожидания Беллмана в том, что во 2-м случае мы находим ценность конкретного состояния **s**, подвергнутого некоторой политике **π**, то есть из изменений в формуле здесь только введение политики **π**

**О чём говорит уравнение ожидания Беллмана?**

# Ожидание Беллмана

**Уравнение Беллмана:**

$$v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

**Уравнение ожидания Беллмана:**

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

Разница между ур-ием Беллмана и ур-ием ожидания Беллмана в том, что во 2-м случае мы находим ценность конкретного состояния  $s$ , подвергнутого некоторой политике  $\pi$ , то есть из изменений в формуле здесь только введение политики  $\pi$

## О чём говорит уравнение ожидания Беллмана?

Приведенное уравнение говорит нам, что ценность конкретного состояния определяется вознаграждением и ценностью следующего состояния, когда мы следуем определенной политике  $\pi$

# Ожидание Беллмана

**Уравнение ожидания Беллмана через Q-функцию:**

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

# Ожидание Беллмана

**Уравнение ожидания Беллмана через Q-функцию:**

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

**О чём говорит уравнение ожидания Беллмана через Q-функцию?**

# Ожидание Беллмана

**Уравнение ожидания Беллмана через Q-функцию:**

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

**О чём говорит уравнение ожидания Беллмана через Q-функцию?**

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

# Ожидание Беллмана

Уравнение ожидания Беллмана через Q-функцию:

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

О чём говорит уравнение ожидания Беллмана через Q-функцию?

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

# Ожидание Беллмана

**Уравнение ожидания Беллмана через Q-функцию:**

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

**О чём говорит уравнение ожидания Беллмана через Q-функцию?**

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

- вознаграждение, которое мы получаем за выполнение определенного действия в состоянии  $s$  и переход в другое состояние  $s'$

# Ожидание Беллмана

Уравнение ожидания Беллмана через Q-функцию:

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

О чём говорит уравнение ожидания Беллмана через Q-функцию?

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

- вознаграждение, которое мы получаем за выполнение определенного действия в состоянии  $s$  и переход в другое состояние  $s'$

# Ожидание Беллмана

**Уравнение ожидания Беллмана через Q-функцию:**

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

**О чём говорит уравнение ожидания Беллмана через Q-функцию?**

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

- вознаграждение, которое мы получаем за выполнение определенного действия в состоянии  $s$  и переход в другое состояние  $s'$
- значение Q-функции для состояния  $s'$  относительно некоторого действия  $a$ , которое наш агент предпримет из состояния  $s$

# Ожидание Беллмана

Уравнение ожидания Беллмана через Q-функцию:

$$q_{\pi}(s, a) = \mathbb{E}_{\pi} [R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

О чём говорит уравнение ожидания Беллмана через Q-функцию?

Из приведенного выше уравнения мы видим, что значение Q-функции может быть разложено на следующие две части:

- вознаграждение, которое мы получаем за выполнение определенного действия в состоянии  $s$  и переход в другое состояние  $s'$
- значение Q-функции для состояния  $s'$  относительно некоторого действия  $a$ , которое наш агент предпримет из состояния  $s$

# **Markov Decision Process (MDP)**

Процесс принятия решений по Маркову

- **Оптимальная Value Function**
- **Оптимальная Q-функция**

# **Оптимальная Value Function**

В процессе принятия решений по Маркову существует множество различных функций ценности (Value Functions) в соответствии с различными политиками (Policy Functions)

# Оптимальная Value Function

В процессе принятия решений по Маркову существует множество различных функций ценности (Value Functions) в соответствии с различными политиками (Policy Functions)

Оптимальной функцией ценности для состояния  $s$  мы будем называть такую Value Function, которая дает максимальное значение по сравнению со всеми другими функциями ценности

# Оптимальная Value Function

В процессе принятия решений по Маркову существует множество различных функций ценности (Value Functions) в соответствии с различными политиками (Policy Functions)

Оптимальной функцией ценности для состояния  $s$  мы будем называть такую Value Function, которая дает максимальное значение по сравнению со всеми другими функциями ценности

**Формула:**

$$v_*(s) = \max_{\pi} v_{\pi}(s)$$

# Оптимальная Q-функция

Аналогично нахождению оптимальной Value Function, оптимальная Q-функция сообщает нам максимальную награду, которую мы собираемся получить, если мы находимся в состоянии **s** и предпринимаем действие **a**

# Оптимальная Q-функция

Аналогично нахождению оптимальной Value Function, оптимальная Q-функция сообщает нам максимальную награду, которую мы собираемся получить, если мы находимся в состоянии **s** и предпринимаем действие **a**

Формула:

$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$$

# Markov Decision Process (MDP)

Процесс принятия решений по Маркову

- **Оптимальная стратегия (Optimal Policy)**

# **Оптимальная стратегия (Optimal Policy)**

**Как понять, что одна стратегия лучше какой-то другой стратегии?**

# Оптимальная стратегия (Optimal Policy)

**Как понять, что одна стратегия лучше какой-то другой стратегии?**

Мы знаем, что для MDP существует стратегия  $\pi$ , лучшая, чем любая другая стратегия  $\pi'$ . Но Как?

# Оптимальная стратегия (Optimal Policy)

**Как понять, что одна стратегия лучше какой-то другой стратегии?**

Мы знаем, что для MDP существует стратегия  $\pi$ , лучшая, чем любая другая стратегия  $\pi'$ . Но Как?

Мы говорим, что стратегия  $\pi$  лучше другой стратегии  $\pi'$ , если Value Function со стратегией  $\pi$  принимает большие значения для всех возможных состояний  $s$ , чем Value Function со стратегией  $\pi'$

# Оптимальная стратегия (Optimal Policy)

**Как понять, что одна стратегия лучше какой-то другой стратегии?**

Мы знаем, что для MDP существует стратегия  $\pi$ , лучшая, чем любая другая стратегия  $\pi'$ . Но Как?

Мы говорим, что стратегия  $\pi$  лучше другой стратегии  $\pi'$ , если Value Function со стратегией  $\pi$  принимает большие значения для всех возможных состояний  $s$ , чем Value Function со стратегией  $\pi'$

**Формула:**  $\pi \geq \pi'$  if  $v_\pi(s) \geq v_{\pi'}(s), \forall s$

# **Оптимальная стратегия (Optimal Policy)**

## **Определение оптимальной стратегии**

Оптимальная стратегия - это та, которая приводит к оптимальной функции ценности (Value Function)

# **Оптимальная стратегия (Optimal Policy)**

## **Определение оптимальной стратегии**

Оптимальная стратегия - это та, которая приводит к оптимальной функции ценности (Value Function)

В MDP может быть более одной оптимальной стратегии. Но все оптимальные стратегии достигают одной и той же оптимальной Value Function и Q-функции

# Оптимальная стратегия (Optimal Policy)

Нахождение оптимальной стратегии

Формула:

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} q_*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

# Оптимальная стратегия (Optimal Policy)

## Нахождение оптимальной стратегии

Формула:

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \underset{a \in \mathcal{A}}{\operatorname{argmax}} q_*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

Мы находим оптимальную стратегию, максимизируя  $\mathbf{q}^*$ , т.е. нашу оптимальную Q-функцию. Затем выбираем то действие **a**, которое дает нам наибольшее возможное значение оптимальной Q-функции

# Оптимальная стратегия (Optimal Policy)

## Нахождение оптимальной стратегии

Формула:

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \underset{a \in A}{\operatorname{argmax}} q_*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

Мы находим оптимальную стратегию, максимизируя  $\mathbf{q}^*$ , т.е. нашу оптимальную Q-функцию. Затем выбираем то действие **a**, которое дает нам наибольшее возможное значение оптимальной Q-функции

Для состояния **s** мы выбираем действие **a** с вероятностью 1, если это действие дает нам максимально возможное значение  $\mathbf{q}^*$ . Поэтому если мы знаем  $\mathbf{q}^*$ , то мы можем получить оптимальную стратегию из этой оптимальной Q-функции

# **Markov Decision Process (MDP)**

Процесс принятия решений по Маркову

- **Уравнения оптимальности Беллмана**
  - **для Value Function**
  - **для Q-функции**

# **Уравнения оптимальности Беллмана**

Уравнения оптимальности Беллмана совпадают с уравнениями ожидания Беллмана, но единственная разница заключается в том, что вместо того, чтобы принимать среднее значение действий (actions), которые может предпринять наш агент, мы принимаем действие с максимальным значением

# Уравнения оптимальности Беллмана

Уравнения оптимальности Беллмана совпадают с уравнениями ожидания Беллмана, но единственная разница заключается в том, что вместо того, чтобы принимать среднее значение действий (actions), которые может предпринять наш агент, мы принимаем действие с максимальным значением

Предположим, что наш агент находится в состоянии  $s$ , и из этого состояния он может выполнить два действия  $a$  и  $a'$ . Итак, мы смотрим на значения Value Function для каждого из этих действий и, в отличие от уравнения ожидания Беллмана, вместо того, чтобы брать среднее значение, наш агент берет такое действие  $a$  или  $a'$ , у которого большее значение  $q^*$

# Уравнения оптимальности Беллмана

Уравнения оптимальности Беллмана совпадают с уравнениями ожидания Беллмана, но единственная разница заключается в том, что вместо того, чтобы принимать среднее значение действий (actions), которые может предпринять наш агент, мы принимаем действие с максимальным значением

Предположим, что наш агент находится в состоянии  $s$ , и из этого состояния он может выполнить два действия  $a$  и  $a'$ . Итак, мы смотрим на значения Value Function для каждого из этих действий и, в отличие от уравнения ожидания Беллмана, вместо того, чтобы брать среднее значение, наш агент берет такое действие  $a$  или  $a'$ , у которого большее значение  $q^*$

**Уравнение оптимальности Беллмана для Value Function:**

$$v_*(s) = \max_a q_*(s, a)$$

# Уравнения оптимальности Беллмана

Уравнение оптимальности Беллмана для Q-функции:

$$q_*(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

# Уравнения оптимальности Беллмана

**Уравнение оптимальности Беллмана для Q-функции:**

$$q_*(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

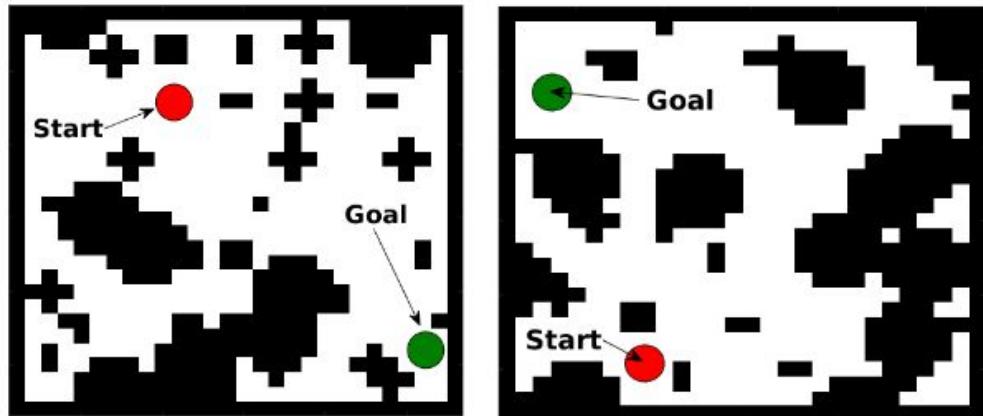
Предположим, наш агент предпринял действие **a** в некотором состоянии **s**. Теперь все зависит от среды, которая может привести нас к любому из других состояний **s'**. Мы по-прежнему берем среднее значение значений обоих состояний, но единственная разница заключается в том, что в уравнении оптимальности Беллмана мы знаем оптимальные значения каждого из состояний

# **Dynamic Programming for RL**

# Dynamic Programming for RL

## Свойства задачи:

1. Оптимальная подструктура (оптимальное решение раскладывается на подзадачи)
2. Перекрывающиеся подзадачи (задача рекуррентно раскладывается на подзадачи, решения подзадач используются повторно для следующих вычислений)



Задача динамического программирования:  
поиск оптимального пути

# Grid World Game

## Задача:

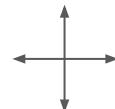
Найти оптимальный путь из любой вершины в **терминальную**

## Формализация задачи:

- за каждое движение не в терминальную вершину награда -1
- не дисконтированный MDP ( $\gamma = 1$ )
- за достижение терминальной вершины награда 0
- за выход с поля награда -1 и возврат в прежнее положение
- агент случайно выбирает направление (политику)

$$\pi(n|\cdot) = \pi(e|\cdot) = \pi(s|\cdot) = \pi(w|\cdot) = 0.25$$

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



возможные действия  
(политика)

# Iterative Policy Evaluation

**Задача:** оценить политику, подсчитав среднюю награду для всех состояний

**Решение:** применение итеративного алгоритма:

- for each iteration  $k + 1$ :
  - for all states  $s \in S$
  - update  $v_{k+1}(s)$  from  $v_k(s')$
  - where  $s'$  is a successor state of  $s$

Уравнение ожидания Беллмана:

$$v_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left( \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_k(s') \right)$$

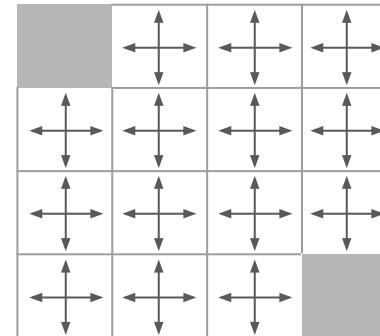
# Iterative Policy Evaluation

$k = 0$

**Value Function**

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

**Policy**



**Важно:** в процессе Policy Evaluation мы **не меняем политику!**

Правая колонка дана **только для демонстрации** создания новой политики.

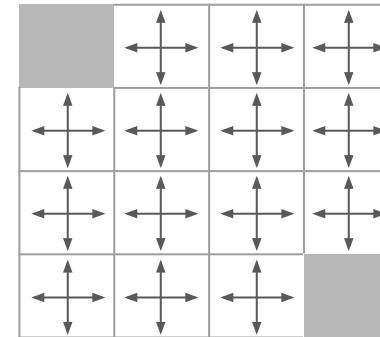
# Iterative Policy Evaluation

$k = 0$

**Value Function**

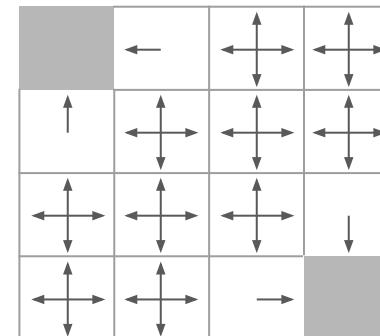
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

**Policy**



$k = 1$

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0



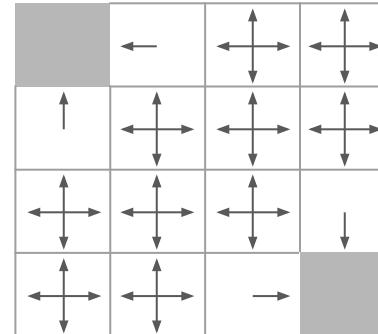
# Iterative Policy Evaluation

$k = 1$

**Value Function**

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

**Policy**



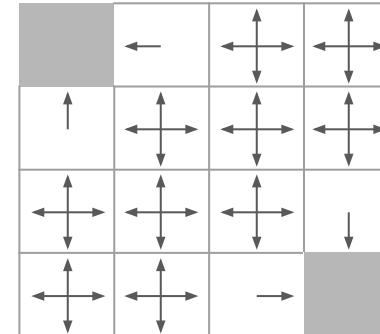
# Iterative Policy Evaluation

$k = 1$

**Value Function**

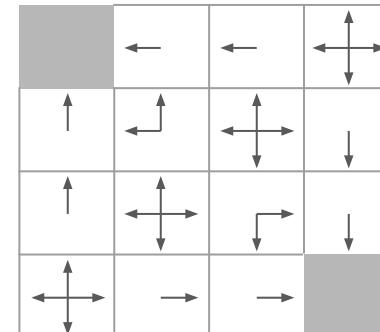
0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

**Policy**



$k = 2$

0	-1.7	-2	-2
-1.7	-2	-2	-2
-2	-2	-2	-1.7
-2	-2	-1.7	0



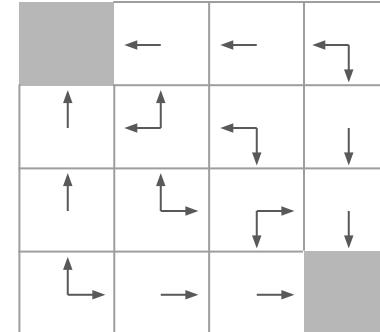
# Iterative Policy Evaluation

$k = 3$

**Value Function**

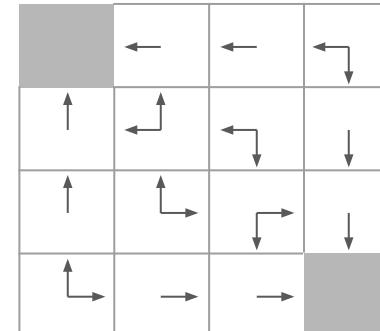
0	-2.4	-2.9	-3
-2.4	-2.9	-3	-2.9
-2.9	-3	-2.9	-2.4
-3	-2.9	-2.4	0

**Policy**



$k = \infty$

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



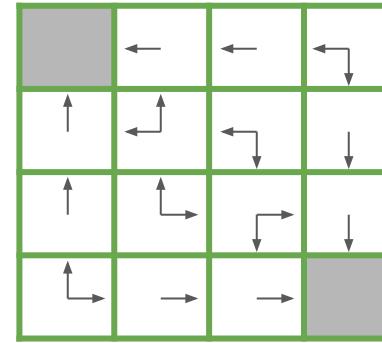
# Iterative Policy Evaluation

$k = \infty$

**Value Function**

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

**Policy**



Значения **Value Function** - какую награду получим в среднем, если будем выбирать путь случайно.  
(сколько в среднем понадобится шагов, чтобы дойти до терминальной позиции)

По окончанию процесса получили **новую оптимальную политику**, из оценки старой (рандомный выбор направления).

# Policy Iteration

**Задача:** найти оптимальную политику

**Решение:** применение итеративного алгоритма:

- Given a policy  $\pi$ :

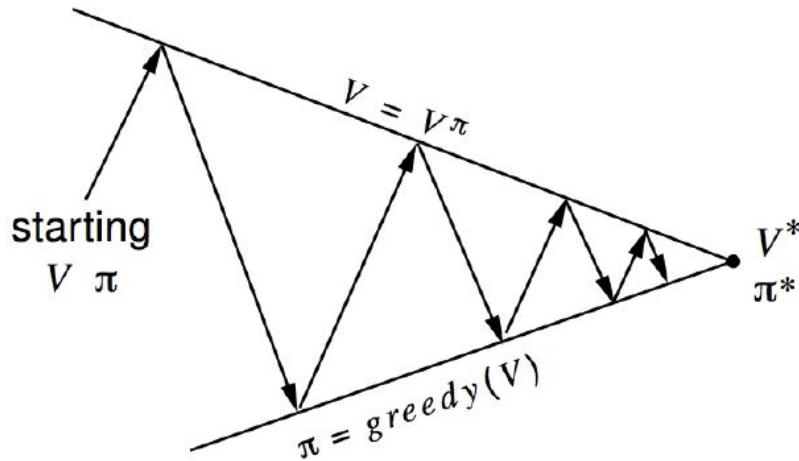
- **Evaluate** the policy  $\pi$

$$v_\pi(s) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \dots | S_t = s]$$

- **Improve** the policy by acting greedily with respect to  $v_\pi$

$$\pi' = \text{greedy}(v_\pi)$$

# Policy Iteration

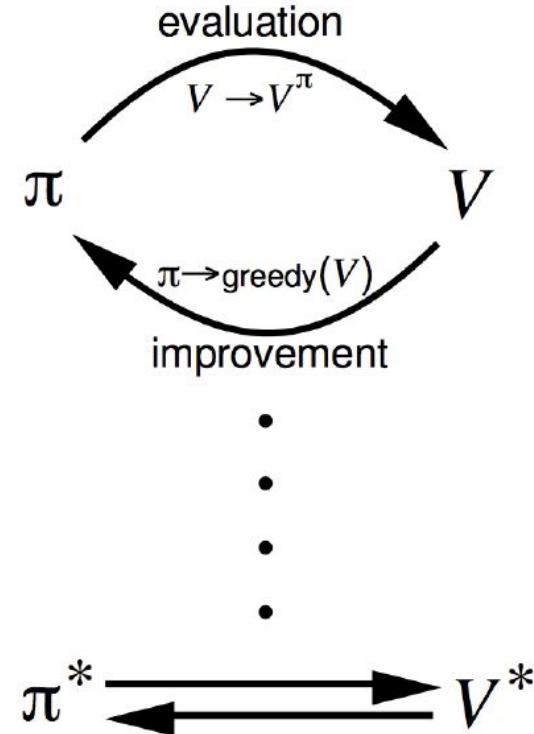


Policy evaluation Estimate  $v_\pi$

Iterative policy evaluation

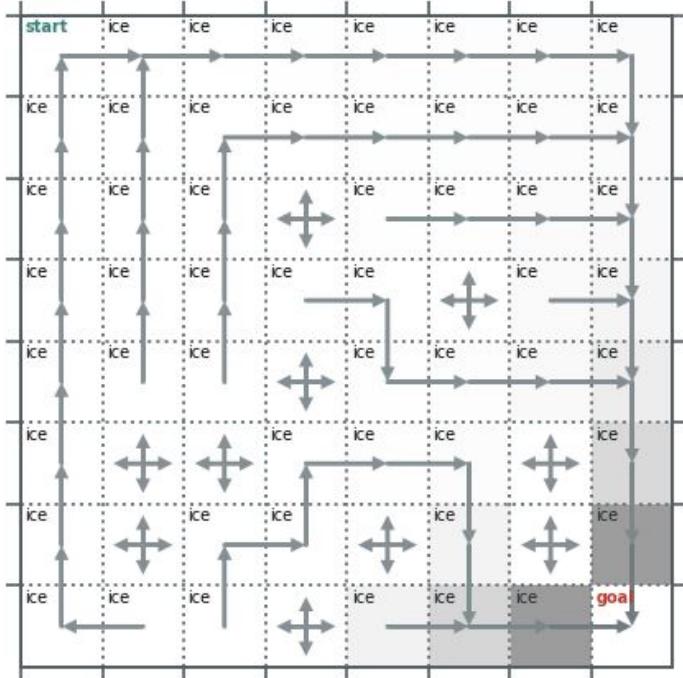
Policy improvement Generate  $\pi' \geq \pi$

Greedy policy improvement

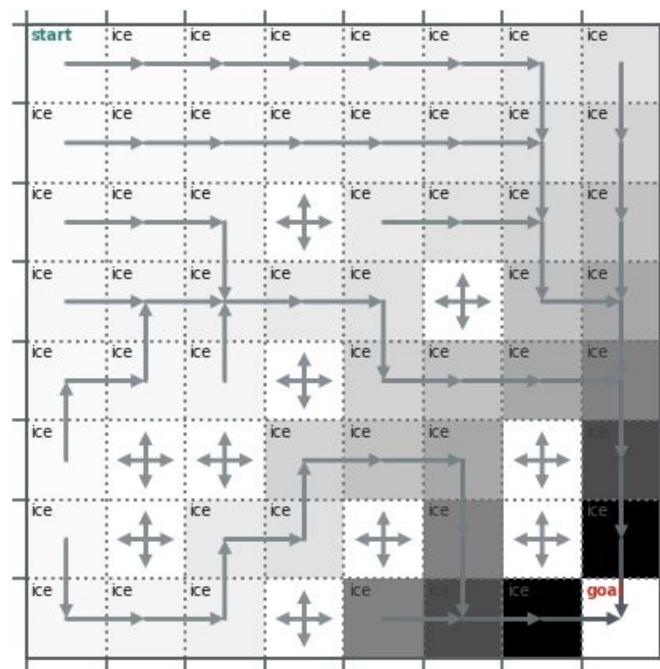


# Policy Iteration

1 итерация



Алгоритм сошёлся



# Policy Improvement

- Consider a deterministic policy,  $a = \pi(s)$
- We can *improve* the policy by acting greedily

$$\pi'(s) = \operatorname{argmax}_{a \in \mathcal{A}} q_\pi(s, a)$$

- This improves the value from any state  $s$  over one step,

$$q_\pi(s, \pi'(s)) = \max_{a \in \mathcal{A}} q_\pi(s, a) \geq q_\pi(s, \pi(s)) = v_\pi(s)$$

- It therefore improves the value function,  $v_{\pi'}(s) \geq v_\pi(s)$

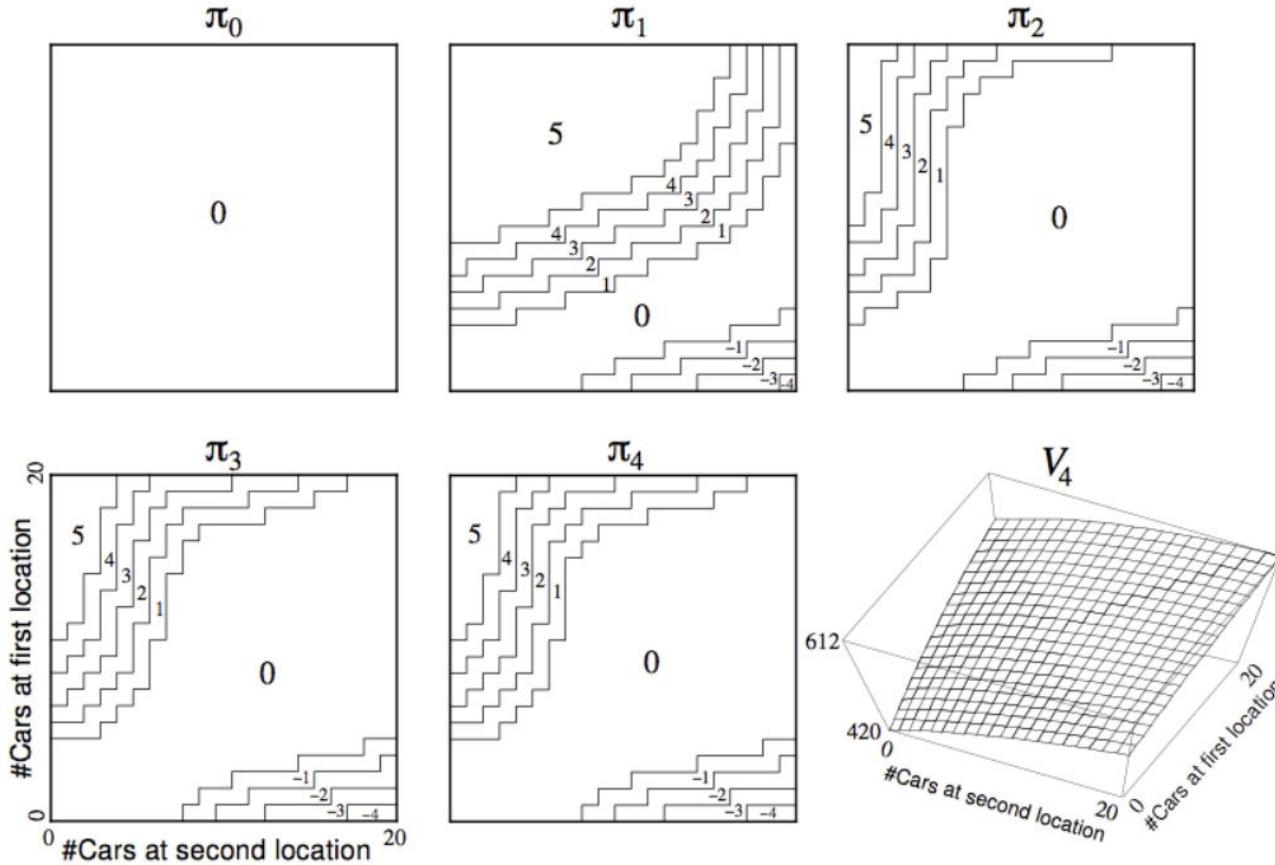
$$\begin{aligned} v_\pi(s) &\leq q_\pi(s, \pi'(s)) = \mathbb{E}_{\pi'} [R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\ &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \mid S_t = s] \\ &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, \pi'(S_{t+2})) \mid S_t = s] \\ &\leq \mathbb{E}_{\pi'} [R_{t+1} + \gamma R_{t+2} + \dots \mid S_t = s] = v_{\pi'}(s) \end{aligned}$$

# Jack's Car Rental



- States: Two locations, maximum of 20 cars at each
- Actions: Move up to 5 cars between locations overnight
- Reward: \$10 for each car rented (must be available)
- Transitions: Cars returned and requested randomly
  - Poisson distribution,  $n$  returns/requests with prob  $\frac{\lambda^n}{n!} e^{-\lambda}$
  - 1st location: average requests = 3, average returns = 3
  - 2nd location: average requests = 4, average returns = 2

# Jack's Car Rental



Источник

# Modified Policy Iteration

1. В Policy Evaluation делаем много холостых итераций до сходимости.  
Может останавливаться раньше?
2. Можно ли делать k-итераций **Policy Evaluation**, а затем обновлять политику?
3. Почему бы не обновлять политику каждую итерацию **Policy Evaluation**?

# Principle of optimality

Оптимальную политику можно разделить на две компоненты:

- Первое оптимальное действие  $A_*$  из  $s$  в  $s'$
- Оптимальная политика действий из  $s'$

## Теорема:

Политика  $\pi(a|s)$  оптимальна для состояния  $s$ ,  $(v_\pi(s) = v_*(s)) \Leftrightarrow$   
 $\pi$  оптимальна  $\forall s'$ , достижимых из  $s$   $(v_\pi(s') = v_*(s'))$

# Value Iteration

**Идея:**

- Знаем решение для подзадачи  $v_*(s')$
- Считаем решение для  $v_*(s)$ , заглядывая на 1 шаг вперед

$$v_*(s) \leftarrow \max_{a \in \mathcal{A}} \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

- Обновляем значения итеративно

# Value Iteration

Пример:

g			

Problem

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

$v_1$

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1

$v_2$

0	-1	-2	-2
-1	-2	-2	-2
-2	-2	-2	-2
-2	-2	-2	-2
-2	-2	-2	-2

$v_3$

0	-1	-2	-3
-1	-2	-3	-3
-2	-3	-3	-3
-3	-3	-3	-3
-3	-3	-3	-3

$v_4$

0	-1	-2	-3
-1	-2	-3	-4
-2	-3	-4	-4
-3	-4	-4	-4
-3	-4	-4	-4

$v_5$

0	-1	-2	-3
-1	-2	-3	-4
-2	-3	-4	-5
-3	-4	-5	-5
-3	-4	-5	-5

$v_6$

0	-1	-2	-3
-1	-2	-3	-4
-2	-3	-4	-5
-3	-4	-5	-6
-3	-4	-5	-6

$v_7$

# Policy Iteration vs Value Iteration

Policy Iteration	Value Iteration
Сложность: $O( A  S ^2 +  S ^3)$ за итерацию	Сложность: $O( A  S ^2)$ за итерацию
Требует меньше итераций	Требует больше итераций

Вывод: на практике используют **Modified Policy Iteration**, регулируя количество итераций в **Policy Evaluation**

# ИСТОЧНИКИ

1. Partical RL (курс от ШАД)
2. Курс на YouTube от DeepMind (англ)
3. Dynamic Programming in RL (Toward Data Science Part 1)
4. Markov Decision Processes and Bellman Equations
5. Reinforcement Learning : Markov-Decision Process (Toward Data Science, part 1)
6. Reinforcement Learning: Bellman Equation and Optimality (Toward Data Science, part 2)