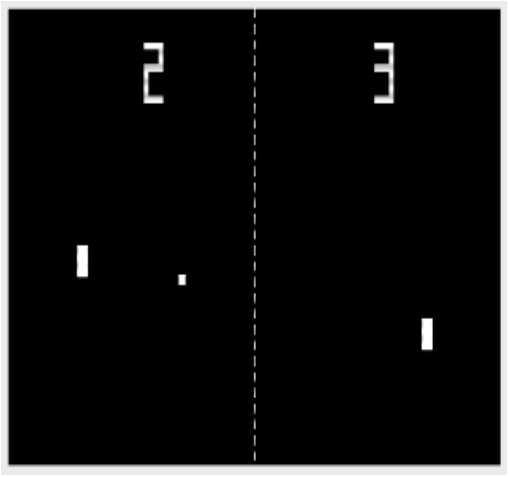
Введение в обучение с подкреплением

Осминин Константин
11 июня 2019

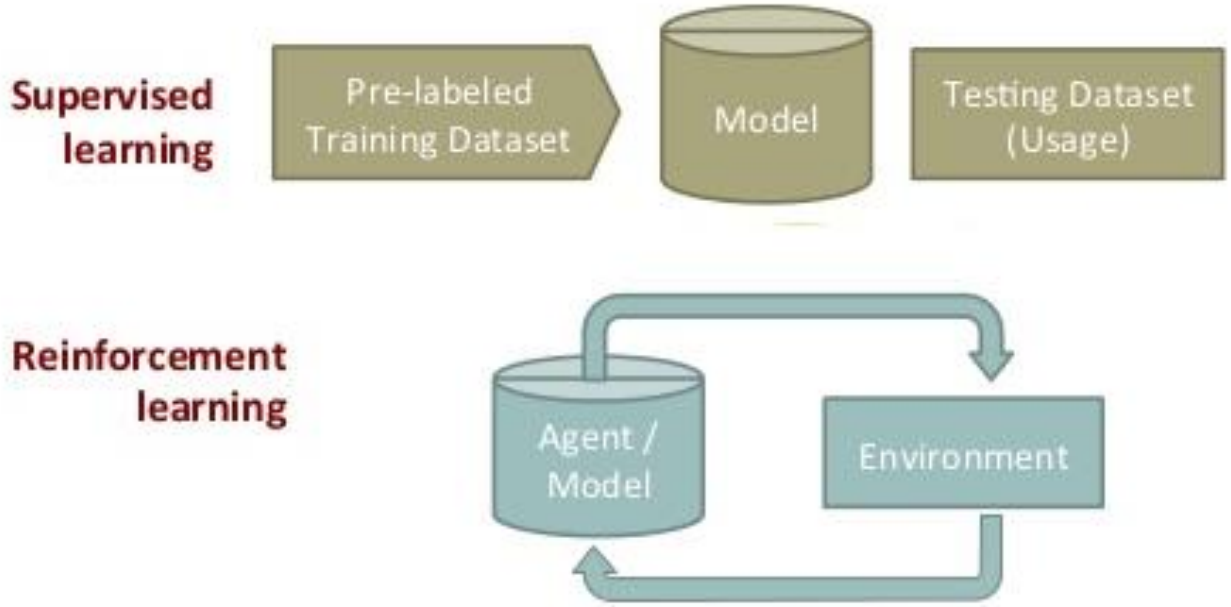
Tinkoff.ru

Как научиться играть в шахматы



- На общий итог влияет последовательность действий
- Отклик на действия может запаздывать
- Многошаговый процесс, ценность каждого действия сложно определить сразу

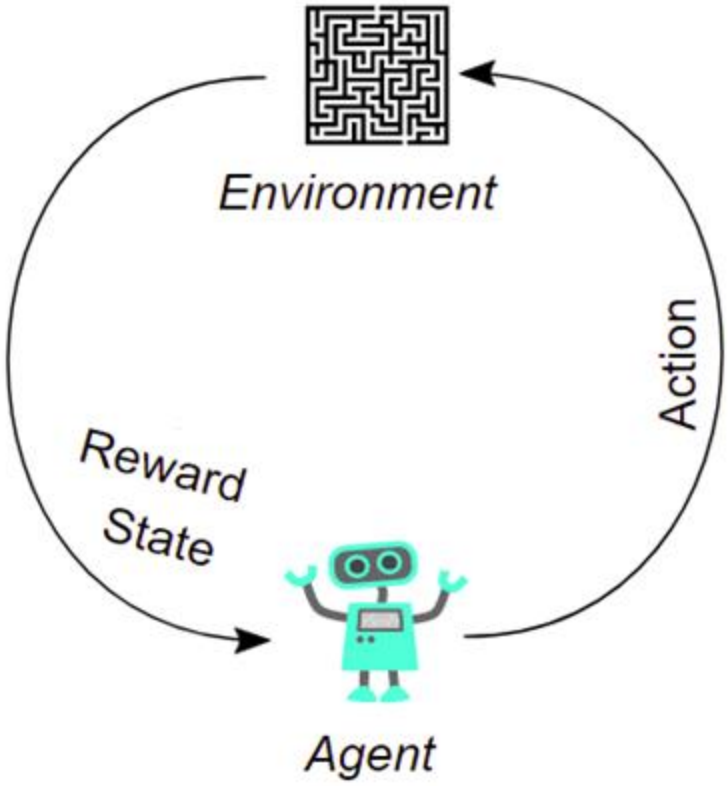
RL vs ML



- Нет учителя, только прямые вознаграждения от среды
- Агент учится определять вклад каждого действия в суммарном вознаграждении
- Агент взаимодействует со средой, учась итеративно

Терминология RL

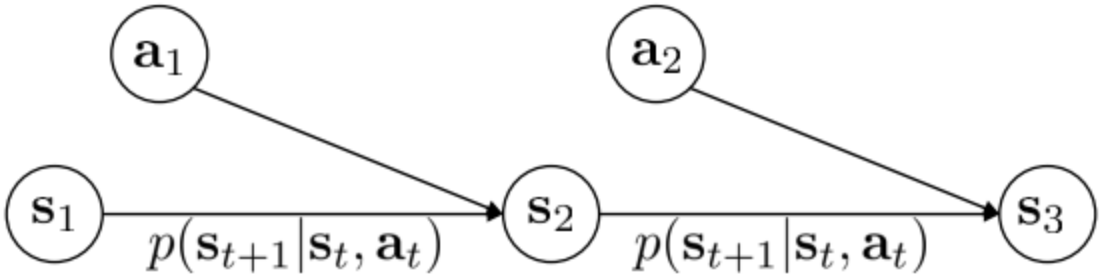
- Мы задаем окружение, или среду (*environment*) – модель реального мира с механикой: правилами, возможными действиями и вознаграждениями.
- Агент взаимодействует с окружением как с черным ящиком, предпринимая действия (*action*) и получая в ответ новое наблюдаемое состояние окружения (*state*) и текущее вознаграждение (*reward*).
- Цикл повторяется до конца игры.
- Цель агента – максимизировать суммарное получаемое вознаграждение.



MDP

Набор $(S, A, p(s'|\, s, a), r(s, a))$ называется Марковским процессом принятия решения (Markov decision process, MDP) , если

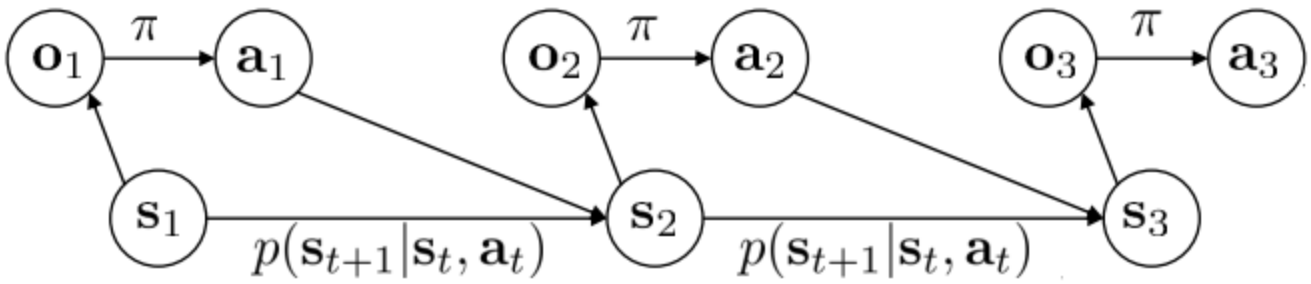
$$p(s_{t+1} = s' \mid s_t, a_t, r_{t-1}, s_{t-1}, a_{t-1}, \dots, r_0, s_0, a_0) = p(s_{t+1} = s' \mid s_t, a_t)$$



POMDP

Набор $(S, \mathrm{O}, A, p(s'|\, s,a), p(o|\, s), r(s,a))$ называется частично наблюдаемым марковским процессом принятия решения (Partial Observable Markov decision process, POMDP) , если

$$p(s_{t+1} = s' \mid s_t, a_t, r_{t-1}, s_{t-1}, a_{t-1}, \dots, r_0, s_0, a_0) = p(s_{t+1} = s' \mid s_t, a_t)$$



Observation vs state

Observation

State

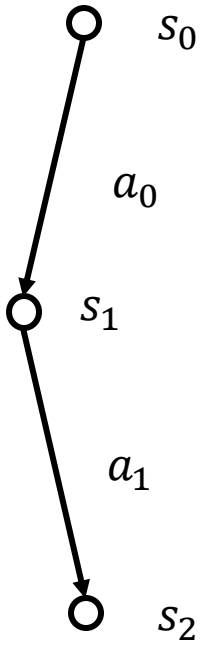
Формальная постановка задачи

$states \;\; s \in S, \;\; actions \;\; a \in A, \; rewards \;\; r \in \mathbb{R}$

Механика среды: $p(s' | \, s, a), \;\; r(s, a)$

Стратегия агента $(policy)$: $\;\; \pi(a \mid s) = p(a \mid s)$

Траектория $\tau = (\, s_0, a_0, r_0, s_1, \, a_1, r_1, s_2, \, a_2, r_2, s_3, , \ldots)$



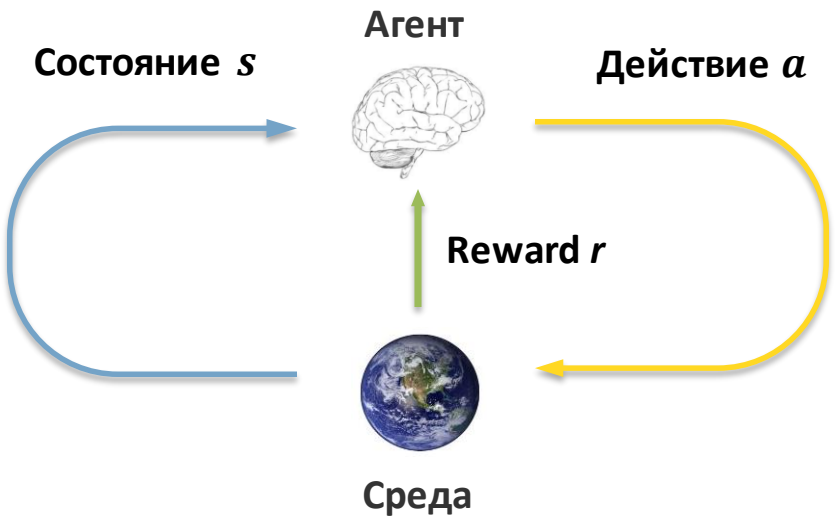
Цель $-$ максимизировать

$$\mathbb{E}_\pi \; R(\tau)$$

$$= \sum_{s_0} p(s_0) \cdot \left(\sum_{a_0} \pi(a_0 \mid s_0) \left(r \;\; (s_0, a_0) + \sum_{s_1} p(s_1 | s_0, a_0) \left(\sum_{a_1} \pi(a_1 \mid s_1)(r \;\; (s_1, a_1) + \cdots) \right) \right) \right)$$

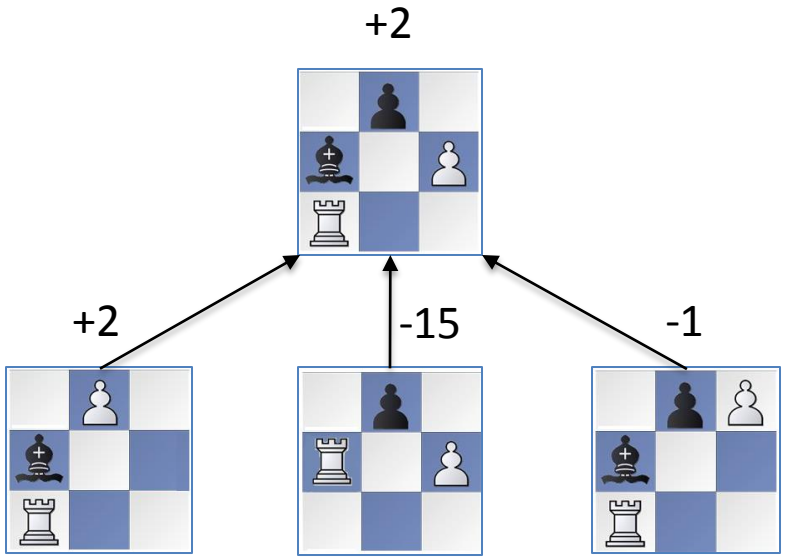
Методы RL

- Имитация
- Value-based
- Policy gradient
- Actor - Critic
- Model-based
- ...



Value based methods

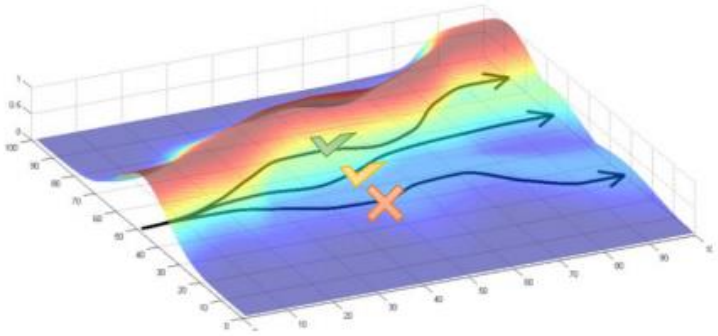
- Ищем ценность состояния $V^{\pi}(s) = \mathbb{E}_{\pi}[R_t \mid s_t = s]$
- Политика тривиальна $\pi \quad (a \mid s) = \begin{cases} 1, a = \underset{a}{\operatorname{argmax}} V(s') \\ 0, \text{иначе} \end{cases}$
- Примеры:
  - Value iteration
  - Q-learning



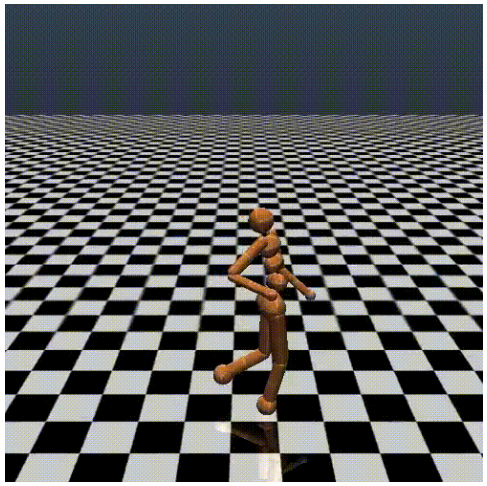
- Pros: Умеет учится на чужом опыте. Гарантии сходимости в некоторых случаях.
- Contras: Оптимизируем не реворд.

Policy gradient methods

- Параметризуем политику $\pi \to \pi_\theta$.
- Градиентный подъем по $\mathbb{E}_{\pi_\theta} R(\tau)$.



- Примеры:
  - REINFORCE
  - Natural gradient
  - PPO. Proximal Policy Optimization
  - TRPO. Trust Region Policy Optimization
- Pros: оптимизируем желаемую метрику *R* напрямую.
- Contras: неустойчиво из-за высокой стохастичности $R(\tau)$. Требовательно к обучающим примерам

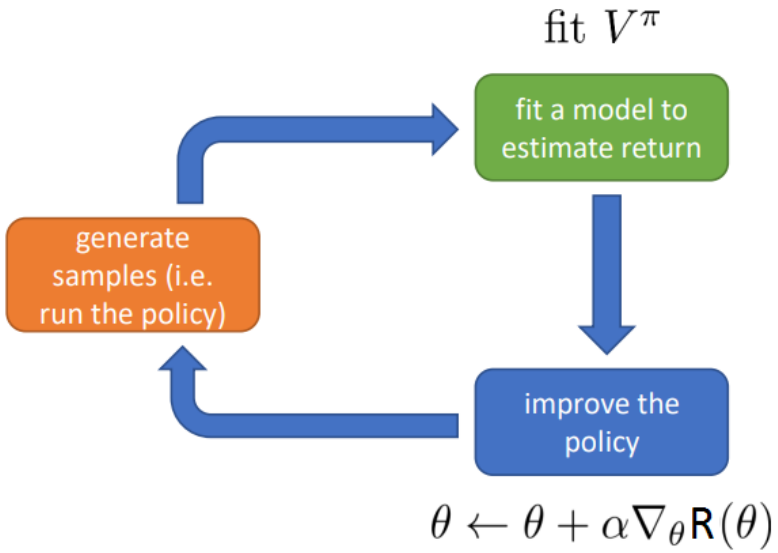


Actor Critic

- Ищем и политику π_θ, и оценку состояния $V_\phi^\pi(s)$.

- Примеры:
  - A3C, Asynchronous Advantage Actor Critic.
  - DDPG, Deep Deterministic Policy Gradient

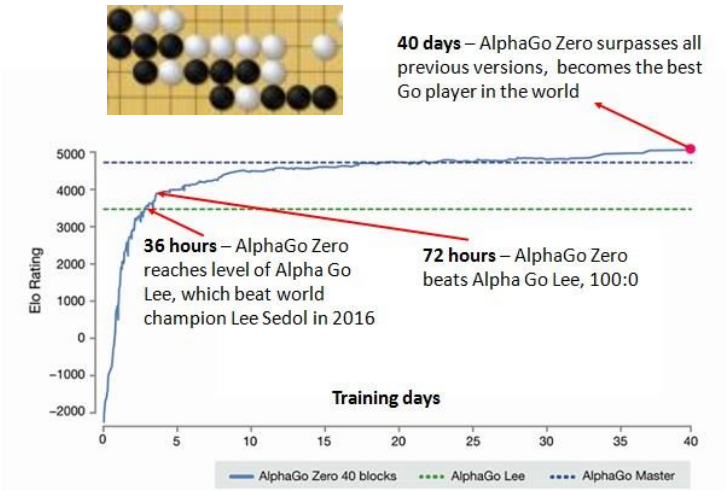
- Pros: Больше устойчивости.
- Contras: Требовательно к обучающим примерам



Model based

- Сначала учим динамику среды $p(s'|\ s, a)$.
- Затем планируем или улучшаем политику

- Примеры:
  - MCTS, Markov Chain Tree Search
  - MPC, Model Predictive Control
  - Dyna-Q

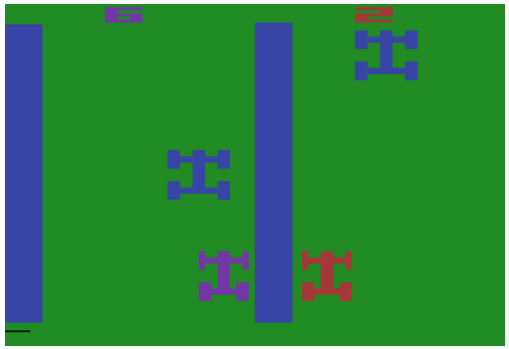


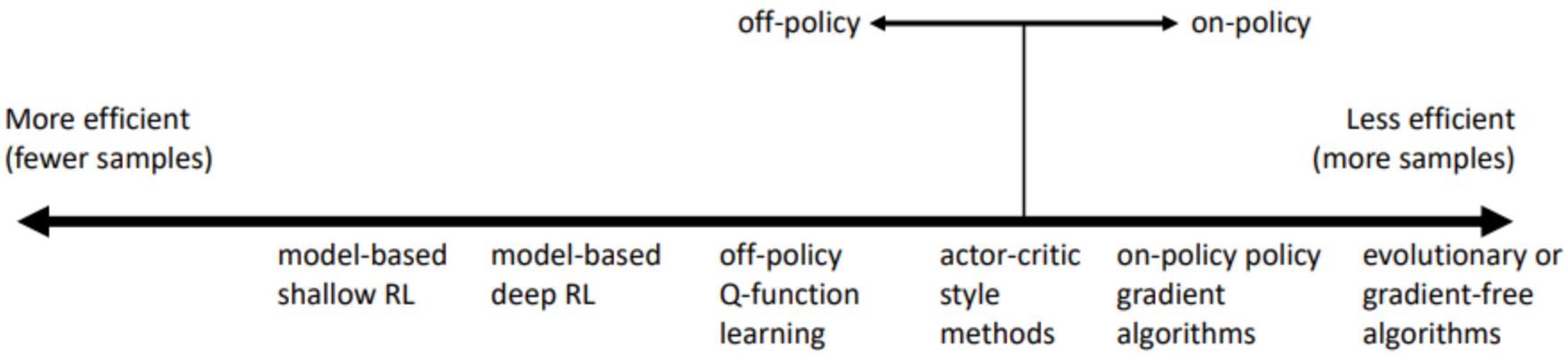
- Pros: Нужно минимум примеров.
- Contras: Можно сильно разойтись с реальностью.

Почему столько алгоритмов ?

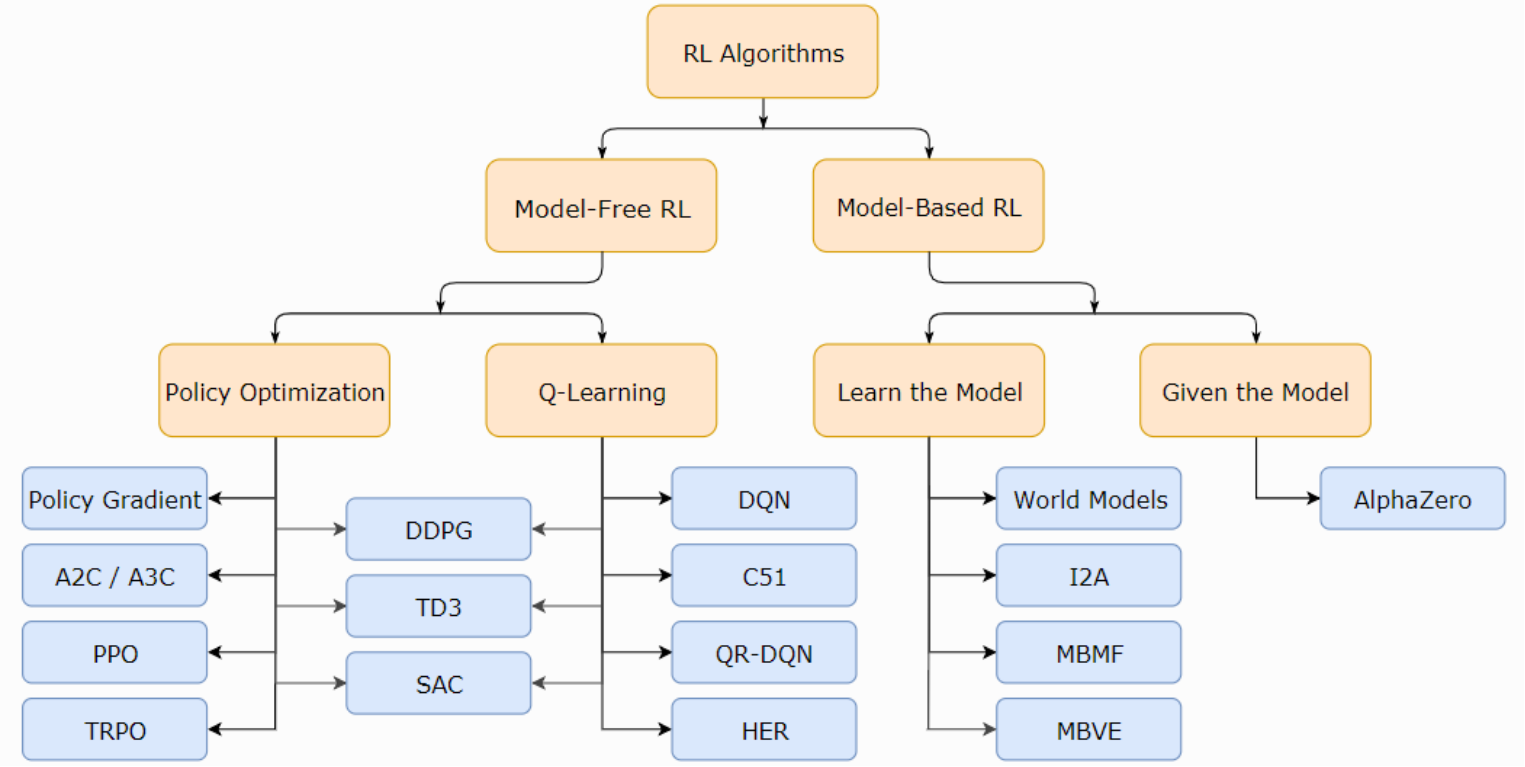
- Требовательность к количеству примеров
- Стабильность обучения
- Знаем ли, как устроена среда
- Дискретные и непрерывные состояния и действия
- Иногда проще моделировать политику, иногда модель

Требовательность к объему примеров





Таксономия методов RL



Source: SpinningUp OpenAI

Структура курса

- **Q-learning .** V и Q значения. Алгоритм. Q(lambda). SARSA. Табличный метод. DQN. Replay buffer. Double Q-learning. Target network.
- **Policy gradient.** REINFORCE. Actor-critic. A2C. A3C. DDPG. PPO. TRPO.
- **Исследование среды**. Эпсилон-жадный алгоритм. Семплирование Томсона. Многорукие бандиты. Оптимизм. Прирост информации.
- **Модельные методы.** Локальные и глобальные методы. LQR.
- **Distributional RL**. Распределенное обучение с подкреплением. Вариационный вывод. Байесовское обучение с подкреплением.
- **Inverse RL.** Восстановление среды по оптимальным траекториям.
- **Transfer learning.** Transfer learning, multitask learning. Meta learning.

Требования курса

- Тервер, матстат
- Машинное обучение, базовые вещи
- Python, Jupyter, pytorch

Литература

- Основы в книге Sutton & Barto Book: Reinforcement Learning: An Introduction
  - complete draft нового издания, на английском
  - первое издание книги, pdf на русском, можно купить бумажную книгу на русском
- Online-курсы и лекции
  - лекции курса CS294 (UC Berkley)
  - материалы курса ШАД по RL
  - OpenAI SpinningUp
  - UC Berkeley RL Bootcamp
  - лекции Дэвида Сильвера (DeepMind)

Домашнее задание

- Семинар ФТШ по pytorch
- Если хочется глубже: 60min blitz tutorial from pytorch

Спасибо