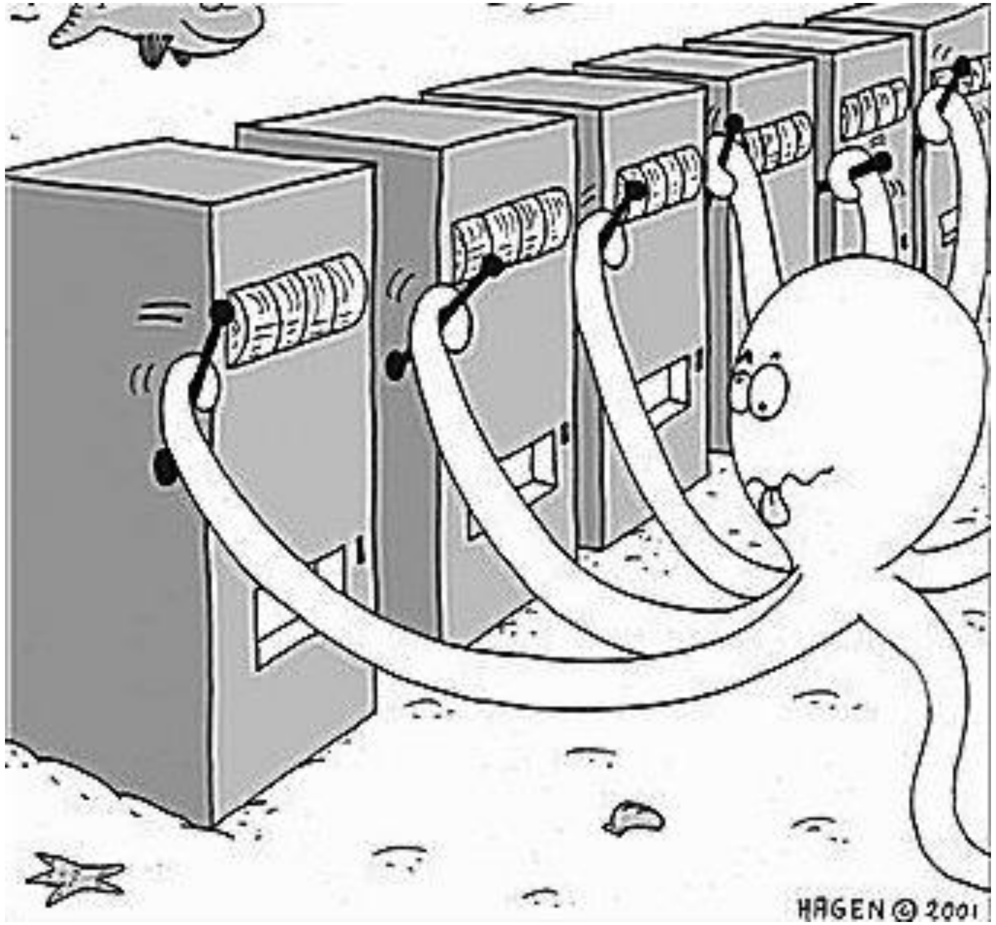
Обучение с подкреплением

Задача о многоруком бандите.



Марковский процесс принятия решений

- S – множество состояний среды
- A - множество действий, которые может совершать агент.
- p(s_{t+1} | s_t, a_t) - Модель перехода среды в новое состояние для текущего состояния среды и действия
- p(s_{r+1} | s_t, a_t) - Модель, описывающая выигрыш для текущего состояния среды и действия
- γ фактор дисконтирования. Контролирует важность будущих событий

Процесс принятия решения - какое действие совершить в текущий момент на основе данных о среде, называется политикой

Детерминированная и стохастическая среда

Агент всегда получает одно и то же вознаграждение за действие при данном состоянии. Среда в свою очередь всегда переходит в одно и то же состояние.

Про повторение одного и того же действия при данном состоянии агент может получать разный выигрыш, а среда – переходить в другое состояние.

- В реальности среда редко будет детерминированной
- Процесс игры происходит следующим образом:

1. Инициализация стратегии pi_0
2. Для всех t = 1,…T
    1. Агент выбирает действие
    2. Среда генерирует премию и новое состояние
    3. Агент корректирует стратегию

Выигрыш: $r_1 + r_2 + \ldots + r_T$

$$total\ discounted\ reward = \sum_{i=1}^{T} \gamma^{i-1} r_i$$

$$V^{\pi}(s) = \mathbb{E}[\sum_{i=1}^{T} \gamma^{i-1} r_i] \quad \forall s \in \mathbb{S}$$

$$V^{*}(s) = \max_{\pi} V^{\pi}(s) \quad \forall s \in \mathbb{S}$$

$$\pi^{*} = \arg\max_{\pi} V^{\pi}(s) \quad \forall s \in \mathbb{S}$$

$$Q : \mathbb{S} \times \mathbb{A} \to \mathbb{R}$$

$$V^*(s) = \max_a Q^*(s, a) \quad \forall s \in \mathbb{S}$$

$$\pi^*(s) = \arg\max_a Q^*(s, a) \quad \forall s \in \mathbb{S}$$

Уравнение Беллмана

$$Q^*(s,a) = R(s,a) + \gamma\,\mathbb{E}_{s'}[V^*(s')]$$

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in \mathbb{S}} p(s'|s,a)V^*(s')$$

Since,

$$V^*(S) = \max_a Q^*(s,a)$$

$$V^*(S) = \max_a \left[R(s,a) + \gamma \sum_{s' \in \mathbb{S}} p(s'|s,a)V^*(s') \right]$$

Value Iteration

Initialize $V(s)$ to arbitrary values

Repeat

    For all $s \in S$

        For all $a \in \mathcal{A}$

            $Q(s,a) \leftarrow E[r|s,a] + \gamma \sum_{s' \in S} P(s'|s,a)V(s')$

        $V(s) \leftarrow \max_a Q(s,a)$

Until $V(s)$ converge

Policy Iteration

Initialize a policy π' arbitrarily
Repeat
    $\pi \leftarrow \pi'$
    Compute the values using π by
        solving the linear equations
            $V^{\pi}(s) = E[r|s, \pi(s)] + \gamma \sum_{s' \in S} P(s'|s, \pi(s)) V^{\pi}(s')$
    Improve the policy at each state
        $\pi'(s) \leftarrow \arg\max_a (E[r|s, a] + \gamma \sum_{s' \in S} P(s'|s, a) V^{\pi}(s'))$
Until $\pi = \pi'$

Exploration and Exploitation

- Достаточно ли у нас знаний о среде, чтобы выбирать оптимальную политику?

- Или существуют еще какие то действия, которые приведут к большему выигрышу?

Eps-жадная стратегия.

- Время от времени будем совершать случайные действия с вероятность eps.
- Чем больше eps тем больше изучаем среду.
- Следует со временем уменьшать eps.

Model-based vs Model-free

- Можно разделить задачи RL на model-based и model-free
- Model-free: ничего не знаем о среде, но нам нужно как то оценивать V(s).
  - Monte Carlo
  - Temporal Difference

$$\cancel{p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}, \quad \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$$

- Model-based: вместо максимизации премии минимизируем стоимость.

$$\underbrace{p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}_{\text{model}}, \quad \cancel{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}$$

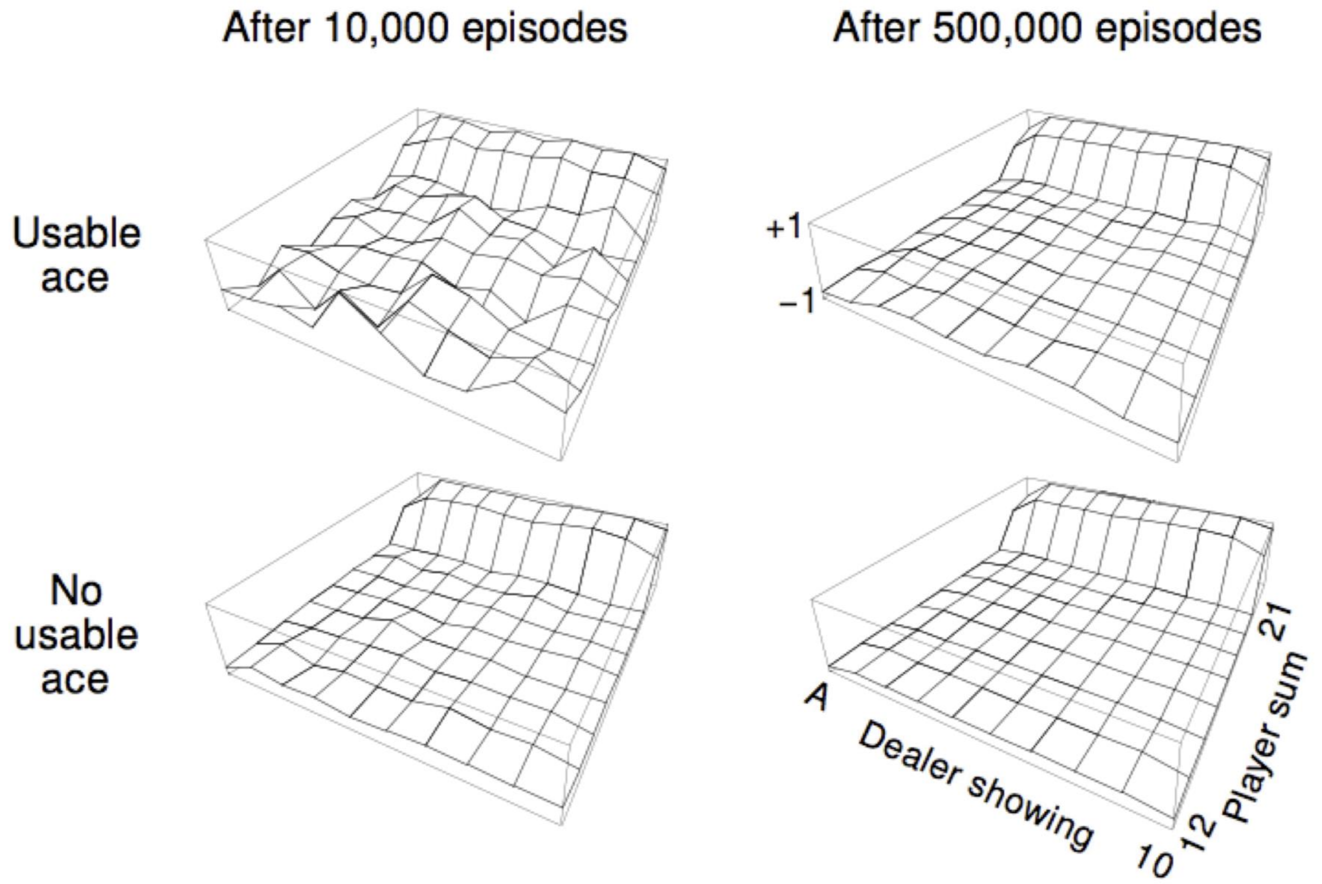
https://medium.com/@jonathan_hui/rl-model-based-reinforcement-learning-3c2b6f0aa323

Blackjack Example

- States (200 of them):
  - Current sum (12-21)
  - Dealer's showing card (ace-10)
  - Do I have a "useable" ace? (yes-no)
- Action stick: Stop receiving cards (and terminate)
- Action twist: Take another card (no replacement)
- Reward for stick:
  - $+1$ if sum of cards $>$ sum of dealer cards
  - 0 if sum of cards $=$ sum of dealer cards
  - -1 if sum of cards $<$ sum of dealer cards
- Reward for twist:
  - -1 if sum of cards > 21 (and terminate)
  - 0 otherwise
- Transitions: automatically twist if sum of cards < 12

http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/MC-TD.pdf

Blackjack Value Function after Monte-Carlo Learning

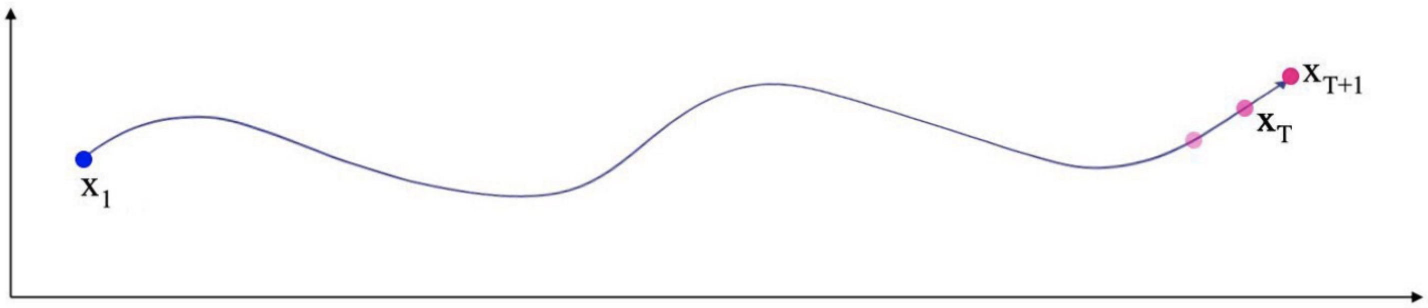


Policy: stick if sum of cards ≥ 20, otherwise twist

Model-based

С функцией стоимости мы ищем оптимальную траекторию

$$\tau = \{\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_T, \mathbf{u}_T\}$$



$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_T} \sum_{t=1}^{T} \underbrace{c(\mathbf{x}_t, \mathbf{u}_t)}_{\text{cost}} \text{ s.t. } \underbrace{\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})}_{\text{model}}$$

Обучение model-based

1. run base policy $\pi_0(\mathbf{a}_t|\mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions

2 шаг – обучение с учителем
3 шаг – оптимизируем траекторию

Функция стоимости показывает, насколько далеко мы находимся от целевого местоположения