

МАРКОВСКИЕ ПРОЦЕССЫ ПРИНЯТИЯ РЕШЕНИЙ

Сергей Николенко

СПбГУ — Санкт-Петербург

15 апреля 2022 г.

Random facts:

- 15 апреля в Грузии — День любви, в Казахстане — День влюблённых (в честь героев эпоса XIII-XIV вв. Козы Корпеш и Баян-Сулу), а в КНДР — Праздник Солнца, то есть день рождения Ким Ир Сена
- 15 апреля — День культуры; 15 апреля 1935 года по инициативе Николая Рериха страны Панамериканского Союза подписали первый международный договор о защите культурного наследия, так называемый пакт Рериха
- 15 апреля 1912 г., в 2:20 ночи, в Атлантическом океане разломился на две части пассажирский пароход «Титаник»
- 15 апреля 1919 г. Совет профсоюзов города Лимерик в Ирландии провозгласил республику Советский Лимерик и стал призывать к всеобщей забастовке; английские власти пошли на некоторые уступки, профсоюз после некоторых колебаний согласился, и 27 апреля республика Советский Лимерик прекратила существование
- 15 апреля 1951 г. Эрик Морли организовал в Лондоне первый конкурс «Мисс мира»; победила Кики Хоканссон из Швеции
- 15 апреля 1955 г. в Дез-Пленз, Иллинойс, Рэй Крок открыл свой первый «Макдоналдс»

КОНКУРЕНТНЫЕ БАНДИТЫ

- Конкурентные бандиты (adversarial bandits) — обобщение многоруких бандитов, в котором награды выбираются не из фиксированных распределений, а выбираются противником (adversary). Т.е. процесс из трёх шагов:
 - противник выбирает распределения наград;
 - агент выбирает ручку, не зная, какие распределения выбрал противник;
 - награда выбирается из соответствующего распределения.
- Здесь есть информационная асимметрия между агентом и противником; без ограничений противник всегда сможет победить любую стратегию, конечно, он выбирает награды $r_j(t)$ произвольно.

- Но сделать что-то можно!
- Можно даже доказать оценки на regret; под regret здесь понимается разница между доходом глобально наилучшей ручки

$$G_{\max}(T) = \max_j \sum_{t=1}^T r_j(t)$$

и ожиданием нашего дохода.

- А идея алгоритма будет в том, чтобы
 - сэмплировать действие с вероятностями $p_1(t), \dots, p_K(t)$ из распределения, которое будет смесью равномерного и экспоненциальных весов от текущей оценки накопленной награды;
 - делать действие, получить награду $r_{i_t}(t)$, но ожидаемую награду считать как $\hat{r}_{i_t}(t) = r_{i_t}(t)/p_{i_t}(t)$; тогда $\mathbb{E}[\hat{r}_{i_t}(t) \mid i_1, \dots, i_{t-1}] = r_{i_t}(t)$.

- Алгоритм Exp3 (Exponential-weight algorithm for Exploration and Exploitation):
 - для данного $\gamma \in [0, 1]$, инициализируем $w_i(1) = 1, i = 1, \dots, K$;
 - на каждом раунде t :
 - считаем вероятности для каждого i :

$$p_i(t) = (1 - \gamma) \frac{w_i(t)}{\sum_{j=1}^K w_j(t)} + \frac{\gamma}{K};$$

- берём следующее действие i_t случайно с вероятностью $p_i(t)$, получаем награду $r_{i_t}(t)$;
- обновляем вес этого действия (остальные не меняются):

$$w_{i_t}(t+1) = w_{i_t}(t) e^{\frac{\gamma}{K} \frac{r_{i_t}(t)}{p_{i_t}(t)}}.$$

- Для такого алгоритма можно доказать adversarial оценку $2.63 \sqrt{G_{\max} K \log K}$ для $\gamma = \min\left(1, \frac{K \log K}{(e-1)g}\right)$, где G_{\max} – верхняя оценка на максимальный доход ручки $\max_j \sum_{t=1}^T r_j(t)$.

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- А что если жизнь устроена сложнее? Пусть мы каждую ручку дёргаем в некотором *контексте*; например, даём рекомендации пользователям, и при этом о пользователях что-то знаем.
- Т.е. на каждом шаге наблюдаем контекст $x_t \in C$, потом выбираем $I_t \in 1, \dots, K$, получаем $r_t \sim p(r_t | I_t, x_t)$.
- Тогда мы уже должны не одну ручку выбрать, а обучить стратегию $\pi : C \rightarrow 1, \dots, K$.

- Наивный подход: давайте запустим Exp3 для каждого контекста по отдельности.
- Это не так уж плохо, если контекстов мало.
- Но получается дополнительный множитель $\sqrt{|C|}$ в оценке на regret, а это нехорошо, вдруг их очень много.

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- Другая идея – давайте вместо этого запускать Exp3 по стратегиям (пусть их мало); на каждом раунде t :
 - получаем совет $\xi_{k,t}$ от каждой стратегии («эксперта») $k \in \Pi$ в виде распределения вероятностей на ручках;
 - берём следующее действие I_t случайно с распределением $p_t = \mathbb{E}_{k \sim q_t} [\xi_{k,t}]$, получаем награду $x_{I_t,t}$;
 - вычисляем ожидаемую награду для каждого i $\tilde{x}_{i,t}$ и ожидаемую награду для каждой стратегии

$$\tilde{y}_{k,t} = \mathbb{E}_{i \sim \xi_{k,t}} [\tilde{x}]_{i,t} = \sum_{i=1}^K \xi_{k,t}(i) \tilde{x}_{i,t}.$$

- обновляем распределение на стратегиях

$$q_{j,t+1} \propto e^{-\eta \sum_{s=1}^t \tilde{y}_{k,s}}.$$

- Это алгоритм Exp4 (Exponential-weight algorithm for Exploration and Exploitation with Experts)
- Всё понятно?..

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- ...но откуда возьмётся $\tilde{x}_{i,t}$?
- Очень важный трюк в reinforcement learning: off-policy estimation.
- Пусть мы хотим оценить качество стратегии π , но играли мы по другой стратегии p . Что делать?
- Нам нужно оценить

$$R(x, \pi(x)) = \frac{1}{n} \sum_{s=1}^n \mathbb{E}_{r \sim p(r|x_s, \pi(x_s))} [r].$$

- Это было бы легко, если бы мы могли оценить для каждой ручки a

$$R(x, a) = \mathbb{E}_{r \sim p(r|a,x)} [r].$$

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- Оказывается, это можно сделать по данным другой стратегии. Рассмотрим

$$\hat{R}(x_s, a) = r_s \frac{[a_s = a]}{p(a_x | x_s)}.$$

- Тут надо только предполагать, что $p(a_x | x_s)$ положительно, т.е. стратегия p покрывает стратегию π .
- Тогда

$$\begin{aligned} \mathbb{E} \left[\hat{R}(x_s, a) \mid x_s, a \right] &= \mathbb{E}_{r_s} \left[\mathbb{E}_{a_s \sim p(x_s)} \left[r_s \frac{[a_s = a]}{p(a_x | x_s)} \right] \right] = \\ &= \mathbb{E}_{r_s} \left[r_s \frac{p(\pi(x_s) = a_s)}{p(a_s | x_s)} \right] = R(x_s, a), \end{aligned}$$

у которого $a_s \sim p$.

- Т.е. надо просто перевзвесить вознаграждения.

• ИТОГО:

Algorithm 2 Exp4 Algorithm (Exponential weights algorithm for Exploration and Exploitation with Experts)

Input: Set of K arms, set of experts Π .

Parameter: real number η

Let q_1 be the uniform distribution over the experts (policies), $\{1, \dots, |\Pi|\}$.

For each round $t = 1, \dots, T$:

- **Receive** expert advice $\xi_{k,t}$ for each expert $k \in \Pi$, where each $\xi_{k,t}$ is a probability distribution over arms.
- **Draw** an arm I_t from the probability distribution $p_t = (p_{1,t}, \dots, p_{K,t})$, where $p_t = \mathbb{E}_{k \sim q_t} \xi_{k,t}$.
- **Observe** loss $\ell_{I_t,t}$. For each arm i , compute $\tilde{\ell}_{i,t}$, using the Inverse Propensity Score trick in Theorem 1 to obtain an unbiased estimator for the loss of arm i :

$$\tilde{\ell}_{i,t} = \frac{\ell_{i,t}}{p_{i,t}} \mathbb{1}_{I_t=i} \quad i = 1, \dots, K$$

- **Compute** the estimated loss for each expert, by taking the expected loss over the expert's predictions.

$$\tilde{y}_{k,t} = \mathbb{E}_{i \sim \xi_{k,t}} \tilde{\ell}_{i,t} = \sum_{i=1}^K \xi_{k,t}(i) \tilde{\ell}_{i,t} \quad k = 1, \dots, |\Pi|$$

- **Compute** the new probability distribution over the experts $q_{t+1} = (q_{1,t+1}, \dots, q_{N,t+1})$, where

$$q_{j,t+1} = \frac{\exp(-\eta \sum_{s=1}^t \tilde{y}_{k,s})}{\sum_{k=1}^{|\Pi|} \exp(-\eta \sum_{s=1}^t \tilde{y}_{k,s})}$$

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- А ещё, конечно, в контекстуальном бандите можно сделать модель какую-нибудь. Например, LinUCB предполагает, что ожидаемая награда – это линейная функция:

$$r_{a,t} = \mathbf{x}_{a,t}^\top \theta_a^* + \epsilon_t,$$

где θ_a^* – вектор коэффициентов, который нужно оценивать, а $\mathbf{x}_{a,t}$ – вектор признаков контекста.

- И тогда по LinUCB надо выбирать $I_t = \arg \max_a u_{a,t}$, где

$$u_{a,t} = \max_{\theta_a \in C_{a,t-1}} \mathbf{x}_{a,t}^\top \theta_a,$$

а $C_{a,t-1}$ – это доверительное множество, аналог доверительного интервала, которое мы в обычном UCB оценивали.

КОНТЕКСТУАЛЬНЫЕ БАНДИТЫ

- Итого:

Algorithm 3 LinUCB with Contextual Bandits

Input: $R \in \mathbb{R}^+$, regularization parameter λ

for $t = 1, 2, \dots, T$ **do**

Observe feature vectors of all arms $a \in \mathcal{A}_t$: $\mathbf{x}_{a,t} \in \mathbb{R}^d$

for all $a \in \mathcal{A}_t$ **do**

if a is new **then**

$\mathbf{A}_a \leftarrow \lambda \mathbf{I}_d$ (d -dimensional identity matrix)

$\mathbf{b}_a \leftarrow \mathbf{0}_{d \times 1}$ (d -dimensional zero vector)

end if

$\hat{\theta}_a \leftarrow \mathbf{A}_a^{-1} \mathbf{b}_a$

$C_{a,t} \leftarrow \left\{ \theta_a^* \in \mathbb{R}^d : \left\| \hat{\theta}_{a,t} - \theta_a^* \right\|_{\mathbf{A}_a} \leq R \sqrt{2 \log \left(\frac{\det(\mathbf{A}_a)^{1/2} \det(\lambda \mathbf{I})^{-1/2}}{\delta} \right) + \lambda^{1/2} S} \right\}$

$p_{a,t} \leftarrow \arg \max_{\hat{\theta}_a \in C_{a,t}} \mathbf{x}_{a,t}^T \hat{\theta}_a$

end for

Choose arm $a_t = \arg \max_{a \in \mathcal{A}_t} p_{a,t}$ with ties broken arbitrarily, and observe payoff r_t

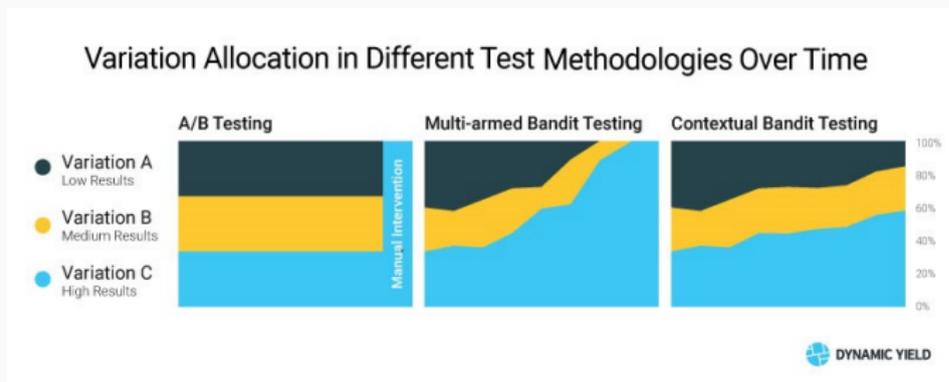
$\mathbf{A}_{a_t} \leftarrow \mathbf{A}_{a_t} + \mathbf{x}_{a_t,t} \mathbf{x}_{a_t,t}^T$

$\mathbf{b}_{a_t} \leftarrow \mathbf{b}_{a_t} + r_t \mathbf{x}_{a_t,t}$

end for

ЗАЧЕМ НУЖНЫ МНОГОРУКИЕ БАНДИТЫ

- Многорукие бандиты используются, например, для A/B тестирования.



- Можно и для оптимизации гиперпараметров в тех же нейронных сетях (или где угодно).
- Контекстуальные – для рекомендаций, для выбора из нескольких вариантов и т.д.
- Но для нас сейчас это первый шаг, упрощённая постановка...

АГЕНТЫ С НЕСКОЛЬКИМИ
СОСТОЯНИЯМИ

- Вернёмся теперь к задаче с несколькими состояниями.
- Вознаграждения (rewards) – на каждом шаге: r_t, r_{t+1}, \dots
- Но что такое «хорошо» in the long run? Как оценивать поведение алгоритма в приведённом выше сеттинге?
- Если есть естественное конечное число шагов (партия), то это *эпизодическая задача* (episodic task), и логично суммировать вознаграждение по эпизоду (до терминального состояния).
- А что с продолжающимися задачами?

- Ваши версии?

- Модель *конечного горизонта*: агент обращает внимание только на следующие h шагов: $E \left[\sum_{t=0}^h r_t \right]$.

- Модель *конечного горизонта*: агент обращает внимание только на следующие h шагов: $E \left[\sum_{t=0}^h r_t \right]$.
- Модель *бесконечного горизонта*: хотелось бы учесть все возможные шаги в будущем, т.к. время жизни агента может быть не определено. Но при этом чем раньше получим прибыль, тем лучше. Как это учесть?

- Модель *конечного горизонта*: агент обращает внимание только на следующие h шагов: $E \left[\sum_{t=0}^h r_t \right]$.
- Модель *бесконечного горизонта*: хотелось бы учесть все возможные шаги в будущем, т.к. время жизни агента может быть не определено. Но при этом чем раньше получим прибыль, тем лучше. Как это учесть?

$$E \left[\sum_{t=0}^{\infty} \gamma^t r_t \right],$$

где γ — некоторая константа (discount factor).

- Модель *конечного горизонта*: агент обращает внимание только на следующие h шагов: $E \left[\sum_{t=0}^h r_t \right]$.
- Модель *бесконечного горизонта*: хотелось бы учесть все возможные шаги в будущем, т.к. время жизни агента может быть не определено. Но при этом чем раньше получим прибыль, тем лучше. Как это учесть?

$$E \left[\sum_{t=0}^{\infty} \gamma^t r_t \right],$$

где γ — некоторая константа (discount factor).

- Модель *среднего вознаграждения* (average-reward model):

$$\lim_{h \rightarrow \infty} E \left[\frac{1}{h} \sum_{t=0}^h r_t \right].$$

ЧТО МЫ БУДЕМ ИСПОЛЬЗОВАТЬ

- Все модели разные, приводят к разным результатам.
- Обычно используется модель бесконечного горизонта с некоторым фиксированным discount factor. Её и мы будем использовать.
- Кроме того, её можно обобщить на эпизодические задачи: достаточно просто положить $\gamma = 1$ и добавить одно лишнее состояние с вознаграждением 0, которое будет замкнуто на себя. Так что отныне навсегда

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}.$$

КАК ОЦЕНИВАТЬ КАЧЕСТВО ОБУЧЕНИЯ?

- Предыдущие модели могут оценивать готовый обучившийся алгоритм. Как оценивать качество обучения?

КАК ОЦЕНИВАТЬ КАЧЕСТВО ОБУЧЕНИЯ?

- Предыдущие модели могут оценивать готовый обучившийся алгоритм. Как оценивать качество обучения?
- Рано или поздно сходится к оптимальному. Это часто можно доказать, но может быть слишком медленно и/или со слишком большими потерями по дороге.

КАК ОЦЕНИВАТЬ КАЧЕСТВО ОБУЧЕНИЯ?

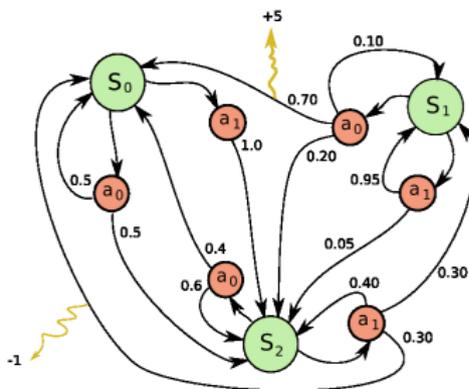
- Предыдущие модели могут оценивать готовый обучившийся алгоритм. Как оценивать качество обучения?
- Рано или поздно сходится к оптимальному. Это часто можно доказать, но может быть слишком медленно и/или со слишком большими потерями по дороге.
- Сходится с большой скоростью. Два подхода:
 - Скорость сходимости к какой-то фиксированной доле оптимальности. Какой?
 - Насколько хорошо себя ведёт алгоритм после фиксированного времени. Какого?

КАК ОЦЕНИВАТЬ КАЧЕСТВО ОБУЧЕНИЯ?

- Предыдущие модели могут оценивать готовый обучившийся алгоритм. Как оценивать качество обучения?
- Рано или поздно сходится к оптимальному. Это часто можно доказать, но может быть слишком медленно и/или со слишком большими потерями по дороге.
- Сходится с большой скоростью. Два подхода:
 - Скорость сходимости к какой-то фиксированной доле оптимальности. Какой?
 - Насколько хорошо себя ведёт алгоритм после фиксированного времени. Какого?
- Минимизировать цену (regret), т.е. уменьшение общей суммы выигрыша по сравнению с оптимальной стратегией с самого начала. Это очень хорошая мера, но результаты о ней получить очень сложно.

МАРКОВСКИЕ ПРОЦЕССЫ

- Марковский процесс принятия решений (Markov decision process) состоит из:
 - множества состояний S ; множества действий A ;
 - функции поощрения $R : S \times A \rightarrow \mathbb{R}$; ожидаемое вознаграждение при переходе из s в s' после $a - R_{ss'}^a$;
 - функции перехода между состояниями $p_{ss'}^a : S \times A \rightarrow \Pi(S)$, где $\Pi(S)$ — множество распределений вероятностей над S .
Вероятность попасть из s в s' после a равна $P_{ss'}^a$.
- Модель марковская: переходы не зависят от истории.



ПОДКРЕПЛЕНИЕ И ЗНАЧЕНИЕ СОСТОЯНИЯ

- Главный момент – разница между *reward function* (непосредственное подкрепление) и *value function* (общее подкрепление, ожидаемое, если начать с этого состояния).



- Суть многих методов обучения с подкреплением – в том, чтобы оценивать и оптимизировать *value function*.

- Для марковских процессов можно формально определить:

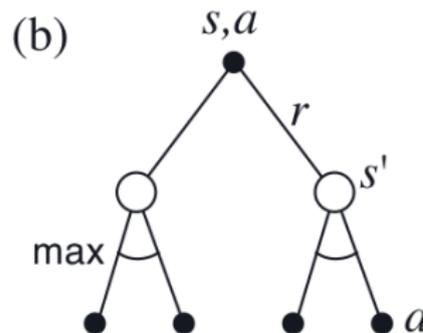
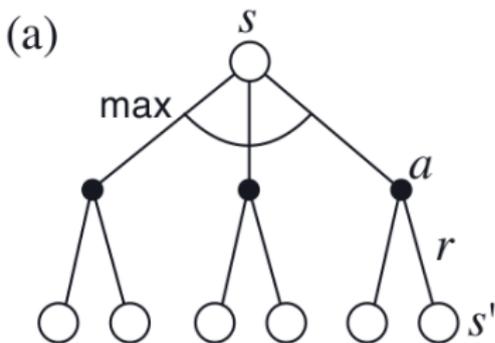
$$V^\pi(s) = \mathbb{E}_\pi [R_t \mid s_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right].$$

- Можно более детализированно – общее подкрепление, ожидаемое, если начать с состояния s и действия a :

$$\begin{aligned} Q^\pi(s, a) &= \mathbb{E}_\pi [R_t \mid s_t = s, a_t = a] = \\ &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right]. \end{aligned}$$

ПОДКРЕПЛЕНИЕ И ЗНАЧЕНИЕ СОСТОЯНИЯ

- Функции V и Q – это как раз то, что нам нужно оценить; если бы мы их знали, можно было бы просто выбирать то a , которое максимизирует $Q(s, a)$.



- Для известной стратегии π V^π удовлетворяют уравнениям Беллмана:

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi [R_t \mid s_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right] = \\ &= \mathbb{E}_\pi \left[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right] = \\ &= \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a \left(R_{ss'}^a + \gamma \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_{t+1} = s' \right] \right) = \\ &= \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V^\pi(s')). \end{aligned}$$

ОСНОВНЫЕ ЗАДАЧИ

- Теоретически всё готово, но у нас много проблем:
 - уравнения знаем, но пока не знаем, как их решать, то есть как найти V^π для данного π ?
 - разных стратегий очень, очень много — как найти оптимальную стратегию поведения агента в данной модели и соответствующие V^* ?
 - но уравнений тоже не знаем — в реальности обычно P и R не даны, их тоже нужно обучить; как?
 - более того, их обычно даже записать не получится, слишком уж много состояний в любой реальной задаче... что делать?



- Давайте есть слона по частям...

ОПТИМАЛЬНЫЕ ЗНАЧЕНИЯ СОСТОЯНИЙ

- *Оптимальное значение состояния* — ожидаемая суммарная прибыль, которую получит агент, если начнёт с этого состояния и будет следовать оптимальной стратегии:

$$V^*(s) = \max_{\pi} E \left[\sum_{t=0}^{\infty} \gamma^t r_t \right].$$

- Эту функцию можно определить как решение уравнений

$$V^*(s) = \max_a \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s')),$$

а затем выбрать оптимальную стратегию

$$\pi^*(s) = \arg \max_a \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s')).$$

- Как решать уравнения?

- Чтобы посчитать value functions для данной стратегии π , можно просто итеративно пересчитывать по уравнениям Беллмана:

$$V^\pi(s) := \sum_a \pi(s, a) \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V^\pi(s')),$$

пока не сойдётся.

- Соответственно, для оптимального надо решать уравнения с максимумами:

$$V^*(s) := \max_a \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V^*(s')).$$

ИТЕРАТИВНОЕ РЕШЕНИЕ (ПО ЗНАЧЕНИЯМ)

- Можно то же самое по Q : пока не сойдётся,

$$Q(s, a) := \sum_{s' \in S} P_{ss'}^a \left(R_{ss'}^a + \gamma \sum_{a'} \pi(s, a') Q(s, a') \right).$$

- А потом просто $V(s) := \max_a Q(s, a)$.
- Оптимальное $Q^*(s, a)$ тоже нехитро:

$$Q^*(s, a) := \sum_{s' \in S} P_{ss'}^a \left(R_{ss'}^a + \gamma \max_{a'} Q^*(s, a') \right).$$

- Пересчёт в предыдущем алгоритме использует информацию от всех состояний-предшественников. Можно сделать другой вариант:

$$Q(s, a) := Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a)).$$

- Он работает, если каждая пара (s, a) встречается бесконечное число раз, s' выбирают из распределения $P_{ss'}^a$, а r сэмплируют со средним $R(s, a)$ и ограниченной дисперсией.
- Но ведь на самом деле нам не V и не Q нужно, а оптимальная стратегия...

- Мы ищем V^π , чтобы улучшить π . Как улучшить π ?
- Естественная идея: давайте жадно выбирать a в s как $\arg \max_a Q^\pi(s, a)$ после вычисления Q^π .
- Policy improvement theorem: для π и π' , если для всех s

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s),$$

то π' не хуже π , т.е. $\forall s V^{\pi'}(s) \geq V^\pi(s)$.

- Как доказать?

- Просто будем разворачивать V^π :

$$\begin{aligned} V^\pi &\leq Q^\pi(s, \pi'(s)) = \mathbb{E}_{\pi'} [r_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_{t+1} = s] \\ &\leq \mathbb{E}_{\pi'} [r_{t+1} + \gamma Q^\pi(s_{t+1}, \pi'(s_{t+1})) \mid s_{t+1} = s] \\ &\leq \dots \leq \\ &\leq \mathbb{E}_{\pi'} [r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots \mid s_{t+1} = s] = V^{\pi'}(s). \end{aligned}$$

ИТЕРАТИВНОЕ РЕШЕНИЕ (ПО СТРАТЕГИЯМ)

- Ищем оптимальную стратегию итеративным алгоритмом.
- **PolicyIteration** – инициализировать π , потом, пока $\pi \neq \pi'$, повторять:

- вычислить значения состояний для стратегии π , решив систему линейных уравнений

$$V^\pi(s) := \sum_a \pi(s, a) \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V^\pi(s'));$$

- улучшить стратегию на каждом состоянии:

$$\pi'(s) := \arg \max_a Q^\pi(s, a) = \arg \max_a P_{ss'}^a (R_{ss'}^a + \gamma V^\pi(s'));$$

- Почему оно сходится?

ИТЕРАТИВНОЕ РЕШЕНИЕ (ПО СТРАТЕГИЯМ)

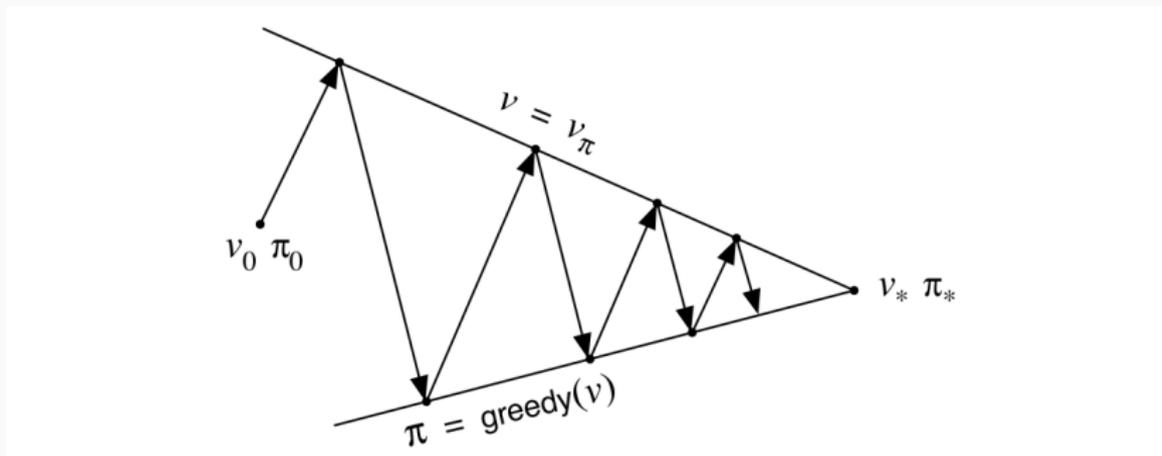
- Сходится, т.к. на каждом шаге строго улучшаем целевую функцию, а всего существует конечное число ($|A|^{|S|}$) стратегий.
- Но, конечно, это медленно, надо V^π пересчитывать; проще делать на каждой итерации ровно один шаг пересчёта V^π , а потом сразу выбирать жадную стратегию:

$$V_{k+1}(s) := \max_a \sum_{s' \in S} P_{ss'}^a (R_{ss'}^a + \gamma V_k(s')).$$

- Это называется value iteration.

ИТЕРАТИВНОЕ РЕШЕНИЕ (ПО СТРАТЕГИЯМ)

- Есть другие похожие методы – их всех объединяет подход, основанный по сути на чём-то вроде EM-алгоритма с динамическим программированием.



- Это может быть достаточно эффективно даже для больших задач (с трюками, позволяющими не всё пространство исследовать).

ОСНОВНЫЕ ЗАДАЧИ

- Теоретически всё готово, но у нас много проблем:
 - уравнения знаем, но пока не знаем, как их решать, то есть как найти V^π для данного π ?
 - разных стратегий очень, очень много — как найти оптимальную стратегию поведения агента в данной модели и соответствующие V^* ?
 - но уравнений тоже не знаем — в реальности обычно P и R не даны, их тоже нужно обучить; как?
 - более того, их обычно даже записать не получится, слишком уж много состояний в любой реальной задаче... что делать?



- Давайте есть слона по частям...

Спасибо за внимание!