DIRECT RL: МОНТЕ-КАРЛО И TD-ОБУЧЕНИЕ

Сергей Николенко

Академия MADE — Mail.Ru

25 сентября 2021 г.

Random facts:

- 25 сентября 1066 г. Гаральд Гардерада вместе с Тостигом Годвинсоном потерпел сокрушительное поражение при Стэмфорд-Бридж, и попытка норвежского завоевания Англии провалилась
- 25 сентября 1493 г. Христофор Колумб отправился в своё второе путешествие к Америке, а 25 сентября 1513 г. Васко де Бальбоа со своим отрядом пересек Панамский перешеек и стал первым европейцем, достигшим Тихого океана
- 25 сентября 1818 г. английский врач Джеймс Бландел впервые провёл операцию по переливанию крови от человека к человеку
- 25 сентября 1962 г. Фидель Кастро заявил, что СССР намерен создать на Кубе базу для своего флота; рыболовного, разумеется
- 25 сентября 1968 г. в первый и пока единственный раз британский хит-парад возглавила русская песня — романс «Дорогой длинною»; правда, называлась она «Those Were the Days» и исполнялась Мэри Хопкин со словами Джина Раскина

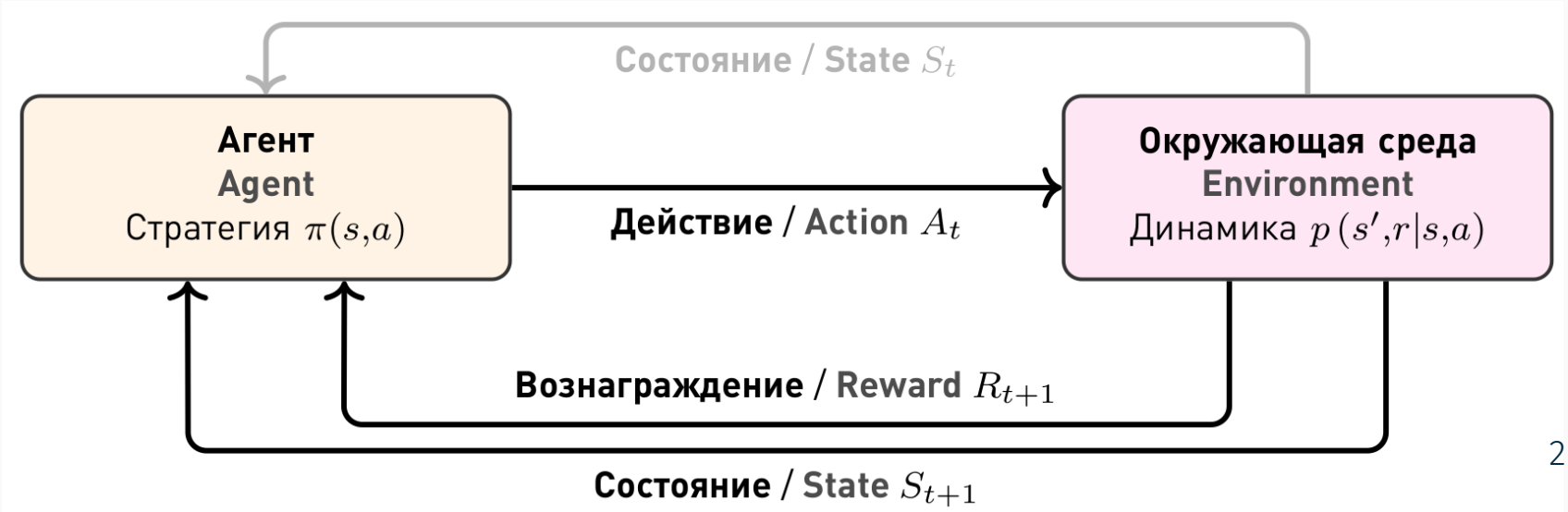
Обучение с подкреплением

- В прошлый раз мы ввели основные понятия динамики марковских процессов принятия решений:
  - собственно динамику процесса:

$$p\left(s', r \mid s, a\right) = p\left(S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a \mid;\right)$$

  - награды за каждый эпизод, начиная со времени t:

$$G_t = R_{t+1} + \gamma G_{t+1} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1};$$



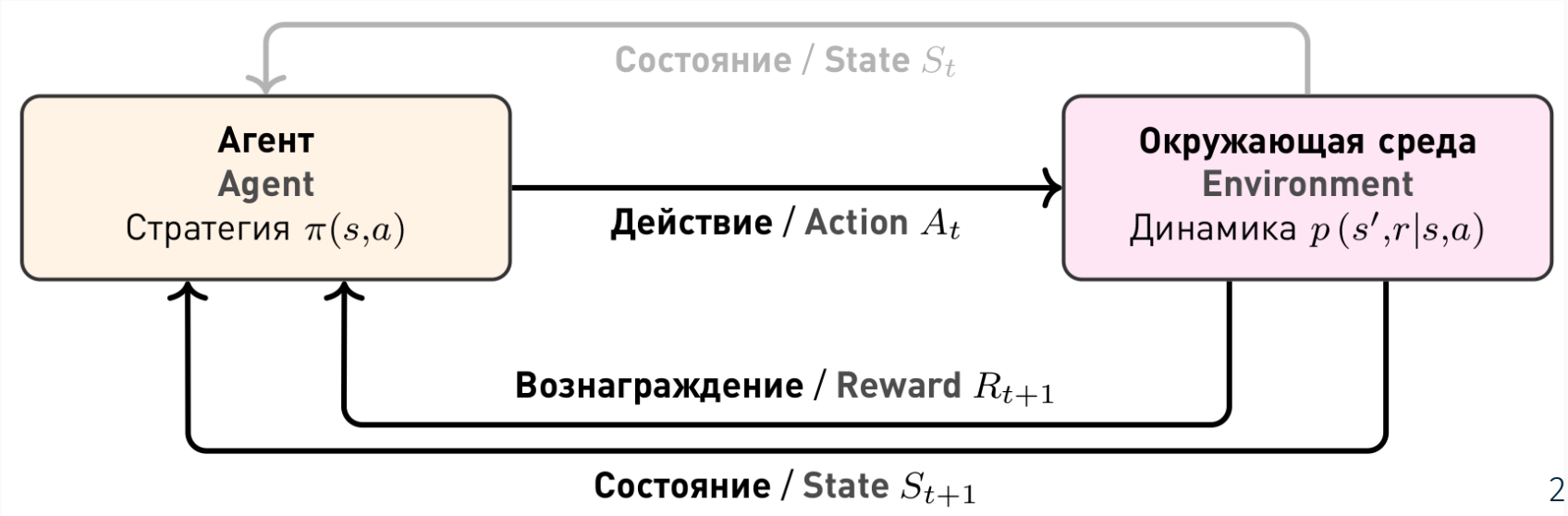
Обучение с подкреплением

- Определили функции значений V и Q

$$V_\pi(s) = \mathbb{E}_\pi\left[G_t \mid S_t = s\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right],$$

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

- Выписали уравнения Беллмана и научились их решать.



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…

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

$$V^{\pi}(s) := \sum_{a} \pi(s,a) \sum_{s' \in S} P^{a}_{ss'} \left(R^{a}_{ss'} + \gamma V^{\pi}(s') \right);$$

  - улучшить стратегию на каждом состоянии:

$$\pi'(s) := \arg\max_{a} Q^{\pi}(s,a) = \arg\max_{a} P^{a}_{ss'} \left(R^{a}_{ss'} + \gamma V^{\pi}(s') \right);$$

- Почему оно сходится?

Итеративное решение (по стратегиям)

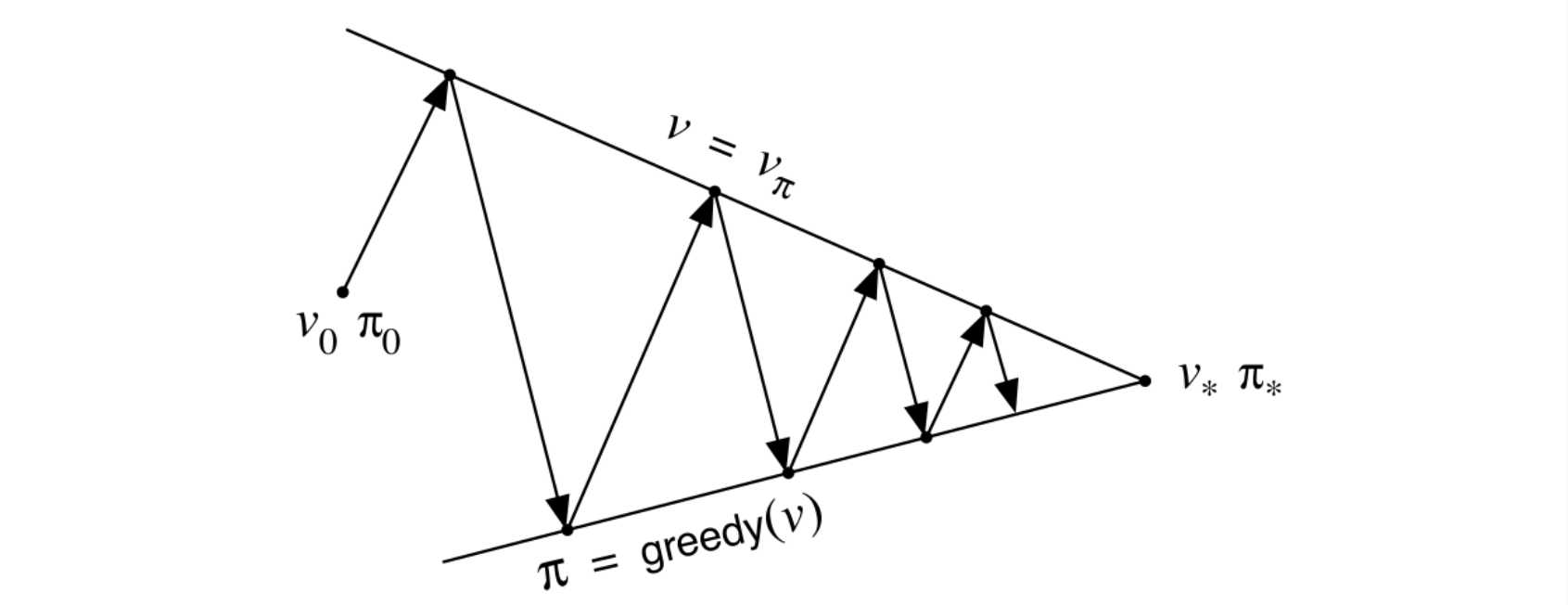
- Сходится, т.к. на каждом шаге строго улучшаем целевую функцию, а всего существует конечное число ($|A|^{|S|}$) стратегий.
- Но, конечно, это медленно, надо V^π пересчитывать; проще делать на каждой итерации ровно один шаг пересчёта V^π, а потом сразу выбирать жадную стратегию:

$$V_{k+1}(s) := \max_a \sum_{s' \in S} P^a_{ss'} \left(R^a_{ss'} + \gamma V_k(s') \right).$$

- Это называется value iteration.

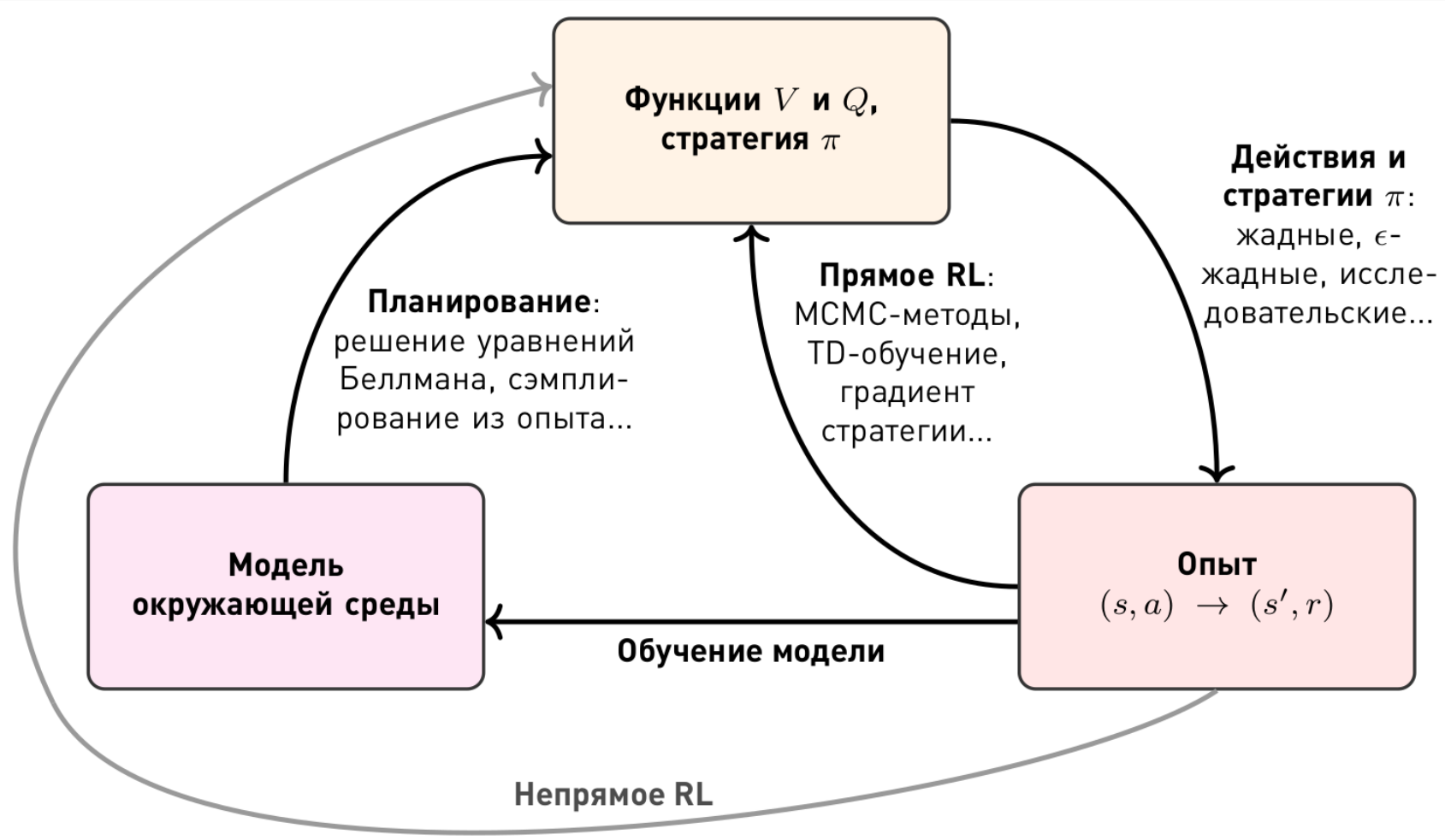
Итеративное решение (по стратегиям)

- 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).

Обучение с подкреплением



Методы Монте-Карло

Метод Монте-Карло

- В прошлый раз мы выписали уравнения Беллмана на V и Q и научились их решать.
- Теперь будем обучать одновременно и модель, и оптимальную стратегию; вознаграждения и переходы не даны.
- Начнём со стохастических алгоритмов (метода Монте-Карло); но начнём опять с простой задачи.
- Как обучить вознаграждения $V^\pi(s)$, ожидаемые от состояния s в эпизодической задаче?

Метод Монте-Карло

- Да очень просто: будем накапливать данные и усреднять.

Алгоритм Monte Carlo estimation:

- инициализировать случайно π и $V(s)$, пустые списки $\mathrm{Ret}(s)$;
- повторять до сходимости:
  - сгенерировать эпизод $S_0, A_0, R_1, S_1, A_1, \dots, S_T$ по стратегии π;
  - $G := 0$
  - для каждого $t = T-1, T-2, \dots, 0$:
    - $G := \gamma G + R_{t+1}$
    - если надо, то добавить G в $\mathrm{Ret}(S_t)$ и обновить $V(S_t) := \mathrm{Avg}(\mathrm{Ret}(S_t))$.

- «Если надо» скрывает тонкую разницу между first-visit и every-visit Monte Carlo.
- На выходе этот алгоритм выдаст V_π для данной π, которой порождаются эпизоды.

Метод Монте-Карло

- Но вообще без модели гораздо удобнее оценивать Q_π. Сразу можно и стратегию обновлять, тот же policy iteration.

Алгоритм Monte Carlo control with exploring starts:

- инициализировать случайно π и $Q(s)$, пустые списки $\mathrm{Ret}(s)$;
- повторять до сходимости:
  - выбрать S_0, A_0 случайно так, чтобы $\forall\ (s,a) \quad p(s,a) > 0$;
  - сгенерировать эпизод $S_0, A_0, R_1, S_1, A_1, \dots, S_T$ по стратегии π;
  - $G := 0$
  - для каждого $t = T-1, T-2, \dots, 0$:
    - $G := \gamma G + R_{t+1}$
    - если надо, то добавить G в $\mathrm{Ret}(S_t, A_t)$ и обновить:

$$Q(S_t, A_t) := \mathrm{Avg}(\mathrm{Ret}(S_t, A_t)),$$
$$\pi(S_t) := \arg\max_a Q(S_t, a).$$

- Этот алгоритм выдаст π_* и соответствующую ей функцию Q_*.
- Здесь важно предположение exploring starts, без него мы не исследуем все действия.

Метод Монте-Карло

- А если оно не выполняется, придётся исследовать самим.

Алгоритм on-policy Monte Carlo control с мягкими стратегиями:

- инициализировать случайно ϵ-мягкую π и $Q(s)$, пустые $\mathrm{Ret}(s)$;
- повторять до сходимости:
  - выбрать S_0, A_0 случайно так, чтобы $\forall\ (s,a) \quad p(s,a) > 0$;
  - сгенерировать эпизод $S_0, A_0, R_1, S_1, A_1, \ldots, S_T$ по стратегии π;
  - $G := 0$
  - для каждого $t = T-1, T-2, \ldots, 0$:
    - $G := \gamma G + R_{t+1}$
    - если надо, то добавить G в $\mathrm{Ret}(S_t, A_t)$ и обновить:

$$Q(S_t, A_t) := \mathrm{Avg}(\mathrm{Ret}(S_t, A_t)),$$
$$a_* := \arg\max_a Q(S_t, a),$$
$$\pi(a \mid S_t) := \begin{cases} 1 - \epsilon + \frac{\epsilon}{|A(S_t)|}, & \text{если } a = a_*, \\ \frac{\epsilon}{|A(S_t)|}, & \text{если } a \neq a_*. \end{cases}$$

- Этот алгоритм умеет искать оптимальную *мягкую* π.

Метод Монте-Карло

- На самом деле для ϵ-мягких стратегий тоже верен аналог policy improvement теоремы, и для ϵ-мягких стратегий метод policy iteration тоже вполне работает.
- Но это on-policy алгоритм, он найдёт мягкую стратегию, а в реальности шахматист, который играет как Магнус Карлсен 90% ходов, а 10% ходов делает случайно, вряд ли продвинется сильно дальше третьего разряда.
- Но ведь и исследовать тоже нужно! Хорошо было бы научиться исследовать по одной стратегии, а оценивать другую...

Метод Монте-Карло

- ...и такой трюк действительно можно сделать!
- Вспомним сэмплирование со значимостями (importance sampling): если мы умеем брать сэмплы по распределению $q(\mathbf{x})$, а оценивать хотим ожидание по распределению $p(\mathbf{x})$, то можно сделать так:

$$\mathbb{E}_{p(\mathbf{x})}\left[f(\mathbf{x})\right] = \int f(\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{x} = \int f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})\mathrm{d}\mathbf{x} = \mathbb{E}_{q(\mathbf{x})}\left[f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}\right].$$

- А у нас в качестве p и q выступают распределения на траекториях:

$$p\left(\mathrm{Traj} \mid \pi\right) = p\left(A_t, S_{t+1}, A_{t+1}, \dots, S_{T-1}, A_{T-1}, S_T \mid \pi\right) =$$
$$= \pi\left(A_t|S_t\right) p\left(S_{t+1} \mid S_t, A_t\right) \dots \pi\left(A_{T-1}|S_{T-1}\right) p\left(S_T \mid S_{T-1}, A_{T-1}\right) =$$
$$= \prod_{k=t}^{T-1} \pi\left(A_k|S_k\right) p\left(S_{k+1} \mid S_k, A_k\right).$$

Метод Монте-Карло

- И получается, что для двух стратегий π, b (от слова behaviour) определить веса

$$\rho_{t:T-1}^{\pi,b} = \frac{p\left(\text{Traj} \mid \pi\right)}{p\left(\text{Traj} \mid b\right)} = \frac{\prod_{k=t}^{T-1} \pi\left(A_k|S_k\right) p\left(S_{k+1} \mid S_k, A_k\right)}{\prod_{k=t}^{T-1} b(A_k \mid S_k) p\left(S_{k+1} \mid S_k, A_k\right)} = \\ = \frac{\prod_{k=t}^{T-1} \pi\left(A_k|S_k\right)}{\prod_{k=t}^{T-1} b(A_k \mid S_k)},$$

  то неизвестные вероятности сократятся и останется, что когда мы порождаем эпизоды по b, нужно просто усреднять не G_t, а $\rho_{t:T-1}^{\pi,b} G_t$, и будут получаться оценки π!
- Единственное условие – покрытие (coverage): должно быть верно, что если $\pi\left(a|s\right) > 0$, то и $b(a \mid s) > 0$.

Метод Монте-Карло

- Ещё есть тонкая разница между обычным importance sampling и взвешенным (weighted importance sampling):
  - в обычном мы берём оценку среднего через сэмплы

$$V(s) = \frac{1}{N} \sum_{t=1}^{N} \rho_{t:T-1}^{\pi,b} G_t,$$

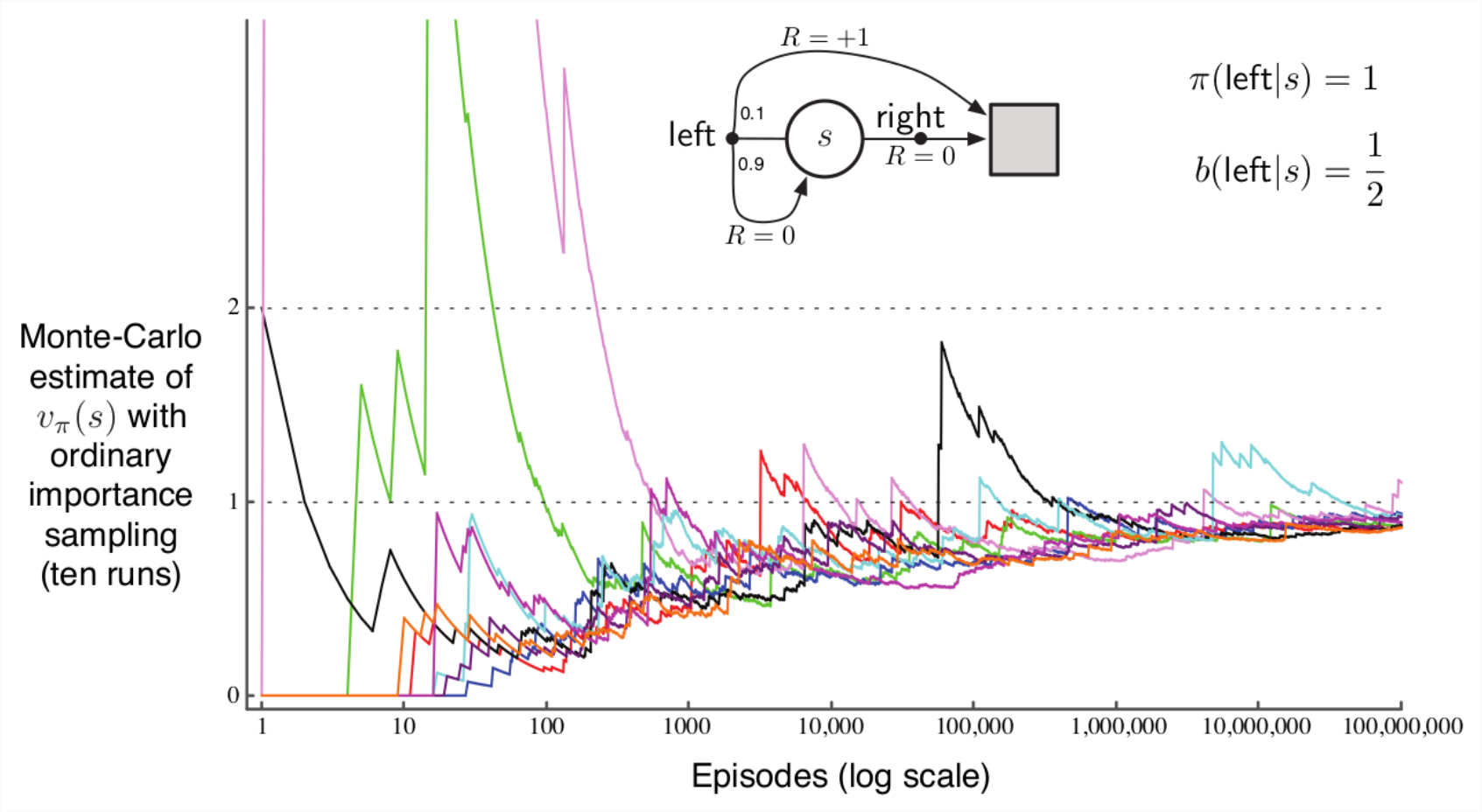
  - а во взвешенном ещё нормируем суммой весов

$$V(s) = \frac{\sum_{t=1}^{N} \rho_{t:T-1}^{\pi,b} G_t}{\sum_{t=1}^{N} \rho_{t:T-1}^{\pi,b}}.$$

- Вторая оценка смещённая, но сходится куда надо и у неё нормальная человеческая дисперсия.

Метод Монте-Карло

- А у первого варианта дисперсия очень большая, и может быть даже бесконечная! Пример:



Метод Монте-Карло

- Итого вот какой алгоритм получается

Алгоритм on-policy Monte Carlo control с мягкими стратегиями:

- инициализировать ϵ-мягкую b, π, $Q(s)$, пустые $\mathrm{Ret}(s)$ и $c(s)$;
- повторять до сходимости:
  - сгенерировать эпизод $S_0, A_0, R_1, S_1, A_1, \dots, S_T$ по мягкой стратегии b;
  - $G := 0$, $W := 1$
  - для каждого $t = T-1, T-2, \dots, 0$:
    - $G := \gamma G + R_{t+1}$
    - $c(S_t, A_t) := c(S_t, A_t) + W$
    - $Q(S_t, A_t) := Q(S_t, A_t) + \frac{W}{c(S_t, A_t)} (G - Q(S_t, A_t))$
    - $\pi(a \mid S_t) := \arg\max_a Q(S_t, a)$
    - если $A_t \neq \pi(S_t)$, то перейти к следующему эпизоду
    - $W := \frac{W}{b(A_t \mid S_t)}$.
- А если убрать $\arg\max$, то получится просто алгоритм оценки данной стратегии π.

TD-обучение

TD-обучение

- Общий принцип TD-обучения: давайте обучать оценки состояний на основе обученных нами ранее оценок для последующих состояний.
- $TD(0)$-обучение: инициализировать $V(s)$ и π произвольно, затем на каждом эпизоде обучения:
  - инициализировать s;
  - для каждого шага t в эпизоде:
    - выбрать A_t в состоянии S_t по стратегии π;
    - сделать A_t, пронаблюдать результат R_{t+1} и следующее состояние S_{t+1};
    - $V(S_t) := V(S_t) + \alpha \left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$.

TD-обучение

- Здесь по сути методы Монте-Карло и TD-обучение расходятся в том, как строить оценку для $V(s)$:
  - MC-методы оценивают $V(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right]$, собирая статистику из G_t;
  - а TD-методы оценивают на один шаг вперёд как $V(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma V_\pi(S_{t+1}) \mid S_t = s \right]$.
- Смысл TD-обучения в том, чтобы использовать уже обученные закономерности для поиска более глубоких закономерностей.
- В результате обучение получится целенаправленным, обучается гораздо быстрее, чем другие стратегии.

TD-обучение

- Здесь тоже есть on-policy и off-policy варианты

Алгоритм Sarsa (on-policy TD control):

- инициализировать случайно $Q(s, a)$;
- повторять до сходимости:
  - инициализировать S_0, выбрать A_0 по стратегии, полученной из Q (например, по ϵ-жадной стратегии);
  - для каждого шага в эпизоде $t = 0, \dots, T$:
    - сделать действие A_t, получить награду R_{t+1}, перейти в состояние S_{t+1};
    - выбрать A_{t+1} по стратегии, полученной из Q (например, по ϵ-жадной стратегии);
    - обновить Q:

$$Q(S_t, A_t) := Q(S_t, A_t) + \alpha \left(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \right)$$

- Этот алгоритм умеет искать оптимальную *мягкую* π (т.е. опять нужно самому исследовать)

TD-обучение

- A off-policy вариант называется Q-обучение; он ещё проще, и это была очень мощная идея, которая до сих пор определяет многое в RL (Watkins, 1989)

Алгоритм Q-learning (off-policy TD control):

- инициализировать случайно $Q(s, a)$;
- повторять до сходимости:
  - инициализировать S_0
  - для каждого шага в эпизоде $t = 0, \dots, T$:
    - выбрать A_t по стратегии, полученной из Q (например, по ϵ-жадной стратегии);
    - сделать действие A_t, получить награду R_{t+1}, перейти в состояние S_{t+1};
    - обновить Q:

$$Q(S_t, A_t) := Q(S_t, A_t) + \alpha \left(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right)$$

- Этот алгоритм умеет искать оптимальную *жёсткую* π_*, делая ходы по мягкой стратегии

Спасибо!

Спасибо за внимание!