2. Khái niệm Off-Policy Learning

2.1. Định nghĩa tổng quát

- **Off-policy learning** là một kỹ thuật trong Reinforcement Learning (RL) mà trong đó **agent học được giá trị (value function) cho một chính sách mục tiêu (target policy)** π, nhưng **bản thân agent lại hành động theo một chính sách khác**, gọi là **behavior policy** b.
  - Tức là:
    - **Behavior policy** b (ký hiệu thường là B) quyết định cách agent tương tác với môi trường, tức xác suất chọn hành động a khi ở trạng thái s, là $b(a|s)$.
    - **Target policy** π là chính sách ta "muốn biết" giá trị của nó; ta sẽ ước lượng hàm giá trị $V^\pi(s)$ hoặc $Q^\pi(s,a)$, nhưng agent không nhất thiết phải thực thi π trong quá trình thu thập dữ liệu.
- Đặc trưng của off-policy learning:
  1. **Behavior policy** → Chịu trách nhiệm "khám phá" (exploration) và sinh dữ liệu (state, action, reward, next state).
  2. **Target policy** → Chính sách mà ta muốn ước lượng giá trị. Có thể đây là chính sách tối ưu (optimal policy) hoặc bất kỳ chính sách nào khác.

2.2. Sự khác biệt so với On-Policy Learning

- **On-Policy Learning**: agent vừa sử dụng chính sách π để **hành động** vừa dùng dữ liệu thu được để **ước lượng/đánh giá** chính sách π. Ví dụ: SARSA, Monte Carlo Prediction với chính sách π.
  - Ưu điểm: đơn giản, không phải "biến đổi" trọng số.
  - Nhược điểm: dễ sa vào khung khám phá hẹp, vì agent chỉ thu thập dữ liệu theo chính sách π mà chính sách này đang thay đổi dần (vừa học vừa đánh giá).
- **Off-Policy Learning**:
  - **Behavior policy** b và **Target policy** π có thể khác nhau hoàn toàn.
  - Agent **hành động** theo b (thường là một chính sách e-greedy hoặc hoàn toàn random để bảo đảm khám phá rộng), nhưng **phiên bản giá trị** (value function) được ước lượng là của π.
  - Ví dụ nổi bật: Q-Learning là off-policy because nó ước lượng $Q^*(s,a)$ (giá trị tối ưu) trong khi hành động có thể dùng e-greedy.

2.3. Tại sao cần Off-Policy Learning?

1. **Khám phá rộng hơn (Better exploration):**
   - Nếu ta chỉ học theo chính sách π (on-policy) ngay từ đầu, π có thể kém (ví dụ chỉ chọn ngẫu nhiên ngẫu nhiên đôi chút), nên agent sẽ chỉ đi vào một vùng trạng thái nhỏ, không bao quát. Với off-policy, ta có thể cho agent theo một behavior policy ưu tiên khám phá, ví dụ random hoàn toàn hoặc e-greedy với epsilon cao, để thu thập dữ liệu phong phú; sau đó ta dùng importance sampling hoặc các công thức off-policy khác để ước tính giá trị cho π.
2. **Tách rời behaviour và target:**
   - Cho phép ta học trước (hoặc song song) đa chính sách khác nhau từ cùng tập dữ liệu. Ví dụ: có thể lưu trữ một replay buffer (trong Deep Q-Learning) và cập nhật Q cho nhiều chính sách π khác nhau.
3. **Học chính sách tối ưu ngay cả khi hành động không hoàn toàn "greedy":**
   - Q-Learning (off-policy) sẽ ước lượng $Q^*(s,a)$ nhưng hành động có thể là e-greedy. Agent không cần thực thi chính sách tối ưu ngay; chỉ cần đảm bảo hành động có xác suất chọn các action đủ để cập nhật giá trị.

2.4. Yêu cầu quan trọng: Phủ (Coverage) của Behavior Policy

- Một **điều kiện bắt buộc** để off-policy learning có thể hội tụ là:

  "Nếu $\pi(a|s) > 0$ (target policy cho phép hành động a ở trạng thái s), thì behavior policy $b(a|s) > 0$ (behavior policy cũng phải có xác suất chọn a ở s lớn hơn 0)."

- Nói cách khác, mỗi cặp (s, a) mà target policy có thể ngỏ (gives non-zero probability) thì behavior policy cũng phải "cover" được, tức không để xác suất của (s, a) bị bằng 0. Nếu không, ta sẽ không bao giờ có dữ liệu cho (s, a), dẫn đến việc ước lượng giá trị $Q^{\pi}(s, a)$ bị sai lệch hoặc không định nghĩa được.
- Ví dụ:
  - Nếu target policy π nói "ở trạng thái s, có 20% chọn action A, 80% chọn action B", thì behavior policy b phải có $b(A|s) > 0$ và $b(B|s) > 0$. Có thể là random 100% (tức $b(A|s) = b(B|s) = 0.5$), hoặc e-greedy,...

---

3. Cách thức hoạt động của Off-Policy Learning

3.1. Giải thích trực quan

- **Target policy** π có thể là "chính sách tối ưu" (optimal policy) hoặc một chính sách cố định mà ta quan tâm. Tuy nhiên, nếu agent chỉ "thử" theo π từ lúc đầu, nó có thể chỉ trải nghiệm một số trạng thái hạn chế (ví dụ: π ưu tiên chọn những action "đã biết" có reward cao, nên agent chỉ đi loanh quanh vùng đó).
- **Behavior policy** b ta chọn để agent thực sự thực thi khi tương tác: mục đích của b là **khám phá**—ví dụ, ta có thể chọn b là chính sách random thuần túy để đảm bảo agent thử càng nhiều action càng tốt, hoặc chọn b là e-greedy với epsilon lớn để cân bằng khám phá.
- Trong quá trình **thu thập dữ liệu**, ta sẽ lưu lại các tuple $(S_t, A_t, R_{t+1}, S_{t+1})$ mà agent gặp phải khi hành động theo b. Sau đó, ta **dùng lại** (reuse) tập dữ liệu đó để ước tính giá trị V^{π} hoặc Q^{π} cho target policy π, bằng cách **trọng số hóa** (weighting) mỗi sample phân phối theo tỉ lệ giữa π và b.

3.2. Ví dụ minh họa khái quát (hình trong slide)

1. **Nếu agent hành động theo target policy** (nếu agent chỉ chạy theo π từ đầu):
   - Agent sẽ có xu hướng "không thử" nhiều trạng thái chưa biết.
   - Kết quả: chỉ có thể ước lượng chính xác cho những trạng thái mà π dẫn agent đến.
2. **Nếu agent hành động theo behavior policy ưu tiên khám phá** (ví dụ random hoặc e-greedy với epsilon cao):
   - Agent sẽ đi qua rất nhiều trạng thái khác nhau, thu thập dữ liệu phong phú.
   - Sau đó, dù π khác b, ta vẫn có thể ước lượng V^{π} hoặc Q^{π} cho π, nhờ cơ chế importance sampling.

---

4. Importance Sampling (Trọng số quan trọng)

Phần quan trọng nhất của off-policy learning là **importance sampling**. Dưới đây là cách giải thích chi tiết:

4.1. Bối cảnh: Đổi mẫu từ phân phối b sang π

- Giả sử ta có một biến ngẫu nhiên X. Nếu X thực sự được "sinh" (sample) từ phân phối b, thì trung bình mẫu (sample average) của X sẽ hội tụ về $E_b[X]$ (kỳ vọng dưới phân phối b).
- Nhưng ta **muốn** ước tính $E_{\pi}[X]$ (kỳ vọng dưới phân phối π). Nếu ta chỉ dùng số mẫu từ b, lấy trung bình thẳng (unweighted average) sẽ ước tính $E_b[X]$, chứ không phải $E_{\pi}[X]$.

Ví dụ minh họa rất đơn giản:

- Giả sử có một đám cầu thủ súc sắc. Phân phối b là "dùng xúc xắc công bằng" (mỗi mặt từ 1 đến 6 đều có xác suất $1/6$). Phân phối π là "xúc xắc lệch" chẳng hạn mặt 6 có xác suất cao hơn. Nếu ta chỉ tung xúc xắc công bằng (theo b) nhiều lần, ta sẽ thu trung bình ~3.5. Nhưng ta muốn biết nếu tung theo π (xúc xắc lệch) thì giá trị trung bình là bao nhiêu. Vì không thể trực tiếp tung xúc xắc lệch, ta có thể tận dụng các "kết quả tung xúc xắc công bằng" rồi trọng số hóa chúng sao cho tương đương với việc tung xúc xắc lệch.

4.2. Định nghĩa tỉ lệ trọng số (Importance Sampling Ratio)

- Cho hai phân phối xác suất rời rạc $b(x)$ (behavior) và $\pi(x)$ (target), với giả thiết "**nếu** $\pi(x) > 0$ thì $b(x) > 0$.** Điều này đảm bảo ta có thể "chuyển" mẫu từ b sang π.
- **Tỉ lệ trọng số** (importance sampling ratio) tại mỗi điểm x được định nghĩa là:

$$\rho(x) = \frac{\pi(x)}{b(x)}.$$

  - Nếu $\pi(x)$ nhỏ hơn $b(x)$, thì $\rho(x) < 1$.
  - Nếu $\pi(x)$ lớn hơn $b(x)$, thì $\rho(x) > 1$.
- Công thức này cho phép ta "chuyển" mỗi giá trị x (được lấy mẫu từ b) thành một mẫu "theo" π, bằng cách gán nó trọng số $\rho(x)$.

4.3. Công thức ước tính kỳ vọng bằng Importance Sampling

Muốn ước tính $E_\pi[X]$, tức:

$$E_\pi[X] = \sum_x \pi(x)\, x,$$

nhưng chỉ có thể lấy mẫu x theo b. Ta viết:

$$\sum_x \pi(x)\, x = \sum_x \underbrace{b(x) \cdot \frac{\pi(x)}{b(x)}}_{\pi(x)}\, x = \sum_x b(x)\, \rho(x)\, x = E_b[\rho(X)\, X].$$

– Tức là: "Kỳ vọng dưới phân phối π" bằng "kỳ vọng dưới phân phối b nhưng có thêm trọng số $\rho(x)$".

Nếu ta có n mẫu độc lập $x_1, x_2, \ldots, x_n$ đều được sinh theo b, thì phép ước tính (estimator) cho $E_\pi[X]$ là:

$$\widehat{E}_\pi[X] = \frac{1}{n} \sum_{i=1}^{n} \rho(x_i)\, x_i.$$

Trong thực tế, nếu biến X là kết quả một dãy hành vi trong RL, thì x_i có thể là một **return từ thời điểm t trở đi** trong một episode, và $\rho(x_i)$ sẽ là tích dãy tỉ lệ giữa π và b ứng với từng action cho đến khi kết thúc.

4.4. Ví dụ minh họa trọng số và cách cập nhật

Trong slide có một loạt hình minh họa quá trình "lấy mẫu và cập nhật giá trị ước tính" theo importance sampling:

1. **Phân phối b (behavior) và π (target):**
   - Giả sử ta có hai phân phối rời rạc trên biến x (có thể $x = 1, 2, 3, \ldots$).
   - Phân phối b là "behavior": chẳng hạn $b(x = 1) = 0.5,\ b(x = 2) = 0.3,\ b(x = 3) = 0.2$.
   - Phân phối π là "target": giả sử $\pi(x = 1) = 0.2,\ \pi(x = 2) = 0.3,\ \pi(x = 3) = 0.5$.
   - Khi ta tung (sample) theo b, độ lệch giữa b và π sẽ được hiệu chỉnh bằng $\rho(x) = \pi(x)/b(x)$.
2. **Quy trình cập nhật ước tính (trace update):**
   - Bắt đầu, ta khởi $\widehat{V} = 0$.
   - Lần 1: Lấy mẫu theo b. Giả sử $x_1 = 1$ (với xác suất 0.5). Tỉ lệ $\rho(x_1) = \pi(1)/b(1) = 0.2/0.5 = 0.4$.
     - Giá trị quan sát tại mẫu này: $x_1 = 1$.
     - Cập nhật ước tính:

$$\widehat{V}^{(1)} = \frac{\rho(x_1)\, x_1}{1} = \frac{0.4 \times 1}{1} = 0.4.$$

   - Lần 2: Lấy mẫu theo b. Giả sử $x_2 = 3$ (với xác suất $b(x = 3) = 0.2$).
     - Tỉ lệ $\rho(x_2) = \pi(3)/b(3) = 0.5/0.2 = 2.5$.

- Quan sát $x_2 = 3$.
- Cập nhật ước tính (theo công thức trung bình trọng số):

$$\widehat{V}^{(2)} = \frac{\rho(x_1)x_1 \ + \ \rho(x_2)x_2}{2} = \frac{0.4 \times 1 \ + \ 2.5 \times 3}{2} = \frac{0.4 \ + \ 7.5}{2} = \frac{7.9}{2} = 3.95.$$

- Tiếp tục như vậy cho các lần lấy mẫu $x_3, x_4, \ldots$. Mỗi lần, ta lấy giá trị x_i theo b, tính $\rho(x_i)$, rồi gộp vào trung bình trọng số để ước tính $E_\pi[X]$.
- Qua hai lần, ta thu được ước tính lần lượt là 0.4 → 3.95. Có thể hình dung rằng ước tính cuối cùng sẽ tiến dần đến giá trị thực $E_\pi[X]$, nằm giữa các giá trị highlight mà slide vẽ trên "một đường thẳng" (giữa hai mốc là giá trị thực, và mỗi update là một "điểm" gần về phía mốc đó).

Ngoài ví dụ đơn giản trên, trong RL ta sẽ mở rộng ý tưởng này:

- Mỗi **episode** (từ thời điểm t đến khi kết thúc) ta tính toán **đoạn return** $G_t = R_{t+1} + \gamma R_{t+2} + \ldots$.
- Nếu hành động tại thời điểm t được chọn theo behavior policy b, nhưng ta muốn ước tính giá trị của target policy π, thì ta tính **tỉ lệ tích luỹ**:

$$\rho_{t:T-1} = \prod_{k=t}^{T-1} \frac{\pi(A_k \mid S_k)}{b(A_k \mid S_k)}.$$

- Mỗi cặp (S_t, A_t) có return tương ứng là G_t, và để cập nhật $Q^\pi(S_t, A_t)$, ta dùng:

$$Q^\pi(S_t, A_t) \leftarrow Q^\pi(S_t, A_t) + \alpha \left[\rho_{t:T-1} \, G_t - Q^\pi(S_t, A_t)\right]$$

  nếu dùng **incremental update** (trong trường hợp khi dùng Monte Carlo off-policy incremental). Hoặc nếu dùng dạng "first-visit/off-policy full-return", ta dùng:

$$ReturnsSum(S_t, A_t) \mathrel{+}= \rho_{t:T-1} \, G_t, \qquad ReturnsCount(S_t, A_t) \mathrel{+}= 1, \qquad Q^\pi(S_t, A_t) = \frac{ReturnsSum(S_t, A_t)}{ReturnsCount(S_t, A_t)}.$$

Trong slide "2.4 Off-policy Learning for Prediction" chỉ trình bày khái niệm cơ bản và ví dụ đơn giản về cách sử dụng trọng số để ước tính kỳ vọng, chứ không đưa ra chi tiết công thức áp dụng cho từng thời điểm trong RL. Nhưng bản chất là **"khi phải ước tính giá trị của target policy π nhưng mẫu lại được sinh theo behavior policy b, thì phải dùng importance sampling ratio để "dịch" các mẫu đó sang π"**.