2. GPI với Temporal Difference

2.1. Khung Generalized Policy Iteration (GPI)

GPI là khuôn khổ lặp giữa hai bước chính:

- 1. **Policy Evaluation** (Đánh giá chính sách): ước lượng hàm giá trị (state-value V^{π} hoặc action-value Q^{π}) cho chính sách hiện tại π .
- 2. **Policy Improvement** (Cải thiện chính sách): xây dựng chính sách mới π' greedy (hoặc ε-greedy) với ước lượng hàm giá trị vừa thu được.

Khi kết hợp, GPI sẽ dần dẫn tới chính sách tối ưu π^* .

2.2. Monte Carlo vs. TD trong GPI

- Monte Carlo: sau mỗi episode hoàn chỉnh mới thực hiện bước Policy Evaluation (ước lượng trung bình return cả episode) rồi mới cải thiện policy.
- TD Learning: có thể đánh giá và cải thiện ngay chỉ sau một bước chuyển (transition) mà không cần chờ tới cuối episode, nhờ bootstrap sử dụng $V(S_{t+1})$ hoặc $Q(S_{t+1}, A_{t+1})$ làm target tạm thời \Box .

Điều này giúp agent cập nhật nhanh hơn, linh hoạt hơn trong quá trình tương tác.

3. Thuật toán SARSA (On-Policy TD Control)

3.1. Định nghĩa SARSA

- SARSA là viết tắt của State-Action-Reward-State-Action.
- Là **thuật toán TD on-policy**, nghĩa là trong quá trình học, agent vừa thực thi chính sách π để tương tác, vừa dùng chính trải nghiệm đó để cập nhật Q^{π} .
- Mục tiêu: ước lượng hàm giá trị hành động $Q^{\pi}(s,a)$ dưới chính sách π , rồi cải thiện dần π cho tới khi hội tụ.

3.2. Các bước thực thi

- 1. Khởi tao
 - Gán giá trị khởi tạo cho tất cả cặp (s, a): Q(s, a) có thể là 0 hoặc số ngẫu nhiên nhỏ.
- 2. Lặp qua mỗi episode (Episodic Interaction)
 - 1. Chọn trạng thái ban đầu S_0 .
 - 2. Chọn hành động đầu A_0 theo chính sách ϵ -greedy dựa trên Q.

- 3. **Với mỗi bước** *t* trong episode cho tới khi kết thúc:
 - Thực thi hành động A_t , quan sát reward R_{t+1} và trạng thái kế tiếp S_{t+1} .
 - Chọn hành động kế tiếp A_{t+1} theo ϵ -greedy dựa trên Q.
 - Cập nhật giá trị $Q(S_t, A_t)$ theo công thức:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)],$$

với

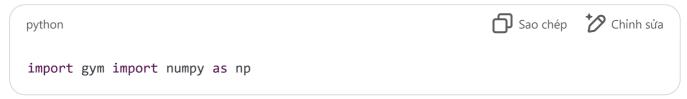
- α là learning rate,
- γ là discount factor,
- $Q(S_{t+1}, A_{t+1})$ là giá trị action-value của bước kế (bootstrap).
- Đặt $t \leftarrow t + 1$ và lặp lại.
- 3. **Kết thúc episode** khi S_{t+1} là trạng thái terminal.

Khi lặp đủ nhiều episode, Q và π (ϵ -greedy với Q) sẽ hội tụ tới hành vi tối ưu

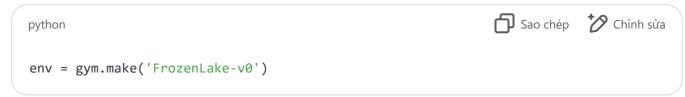
4. Ví dụ minh họa với môi trường FrozenLake

Slide có ví dụ code trên môi trường FrozenLake-v0 của thư viện Gym. Quy trình chung:

1. Import thư viện:



2. Khởi tạo môi trường:



- 3. Thiết lập tham số:
 - Số episode, độ dài tối đa mỗi episode, α , γ , ε .
- 4. Hàm chọn hành động ϵ -greedy dựa trên Q.
- 5. Vòng lặp huấn luyện:
 - ullet Với mỗi episode: reset môi trường, chọn A_0 , sau đó thực hiện SARSA update cho tới khi xong.
- 6. Đánh giá:

• Chạy nhiều episode với policy greedy (không khám phá) để tính tỷ lệ thành công (win rate), đánh giá hiệu quả học.

Ví dụ này cho thấy SARSA on-policy TD control có thể học dần cách di chuyển an toàn trên bề mặt băng (FrozenLake) sao cho tránh hố và đạt tới đíc