

Dạy máy học cách tự chơi



Làm thế nào chúng ta dạy một cỗ máy chơi cờ vua mà không cần cung cấp cho nó một "sách quy tắc" về các nước đi hay?

Học có giám sát (*Supervised learning*) cần các ví dụ được gán nhãn (ví dụ: ‘ở vị trí này, hãy đi nước này’). Nhưng trong nhiều trường hợp, phản hồi như vậy không có sẵn.

Thay vào đó, tác nhân (*agent*) chỉ nhận được phản hồi sau một chuỗi dài các hành động. Ví dụ, nó chỉ biết mình đã làm tốt khi (vô tình) chiếu hết đối thủ.

Loại phản hồi này, nhận được vào cuối trò chơi hoặc trong quá trình thực hiện, được gọi là **phản thưởng (reward)** hay **sự củng cố (reinforcement)**.

Học Tăng Cường: Học cách hành động từ thành công và thất bại

Định nghĩa: Nhiệm vụ của học tăng cường là sử dụng các phần thưởng quan sát được để học một **chính sách (policy)** tối ưu cho môi trường. Tác nhân không có kiến thức tiên nghiệm về cách môi trường hoạt động.

Các thành phần cốt lõi:

- **Tác nhân (Agent):** Thực thể học hỏi và ra quyết định.
- **Môi trường (Environment):** Thế giới mà tác nhân tương tác.
- **Trạng thái (State), s:** Một mô tả về môi trường tại một thời điểm.
- **Hành động (Action), a:** Lựa chọn mà tác nhân có thể thực hiện.
- **Phản thưởng (Reward), R(s):** Phản hồi từ môi trường.

Mục tiêu: Học một chính sách $\pi(s) \rightarrow a$ để tối đa hóa tổng phần thưởng kỳ vọng trong tương lai.

Khung chính thức: Vấn đề này thường được mô hình hóa dưới dạng **Quy trình Quyết định Markov (Markov Decision Process - MDP)**.



Mục tiêu không chỉ là phần thưởng tức thời



Một chính sách tốt không chỉ tối đa hóa phần thưởng tiếp theo, mà là tổng phần thưởng kỳ vọng trong tương lai (có chiết khấu).

Chúng ta định nghĩa **Hữu dụng (Utility)** của một trạng thái, $U(s)$, là tổng phần thưởng (chiết khấu) kỳ vọng nếu tác nhân bắt đầu từ trạng thái s và tuân theo chính sách π .

$$U^\pi(s) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t) \right] \text{ trong đó } S_0 = s$$

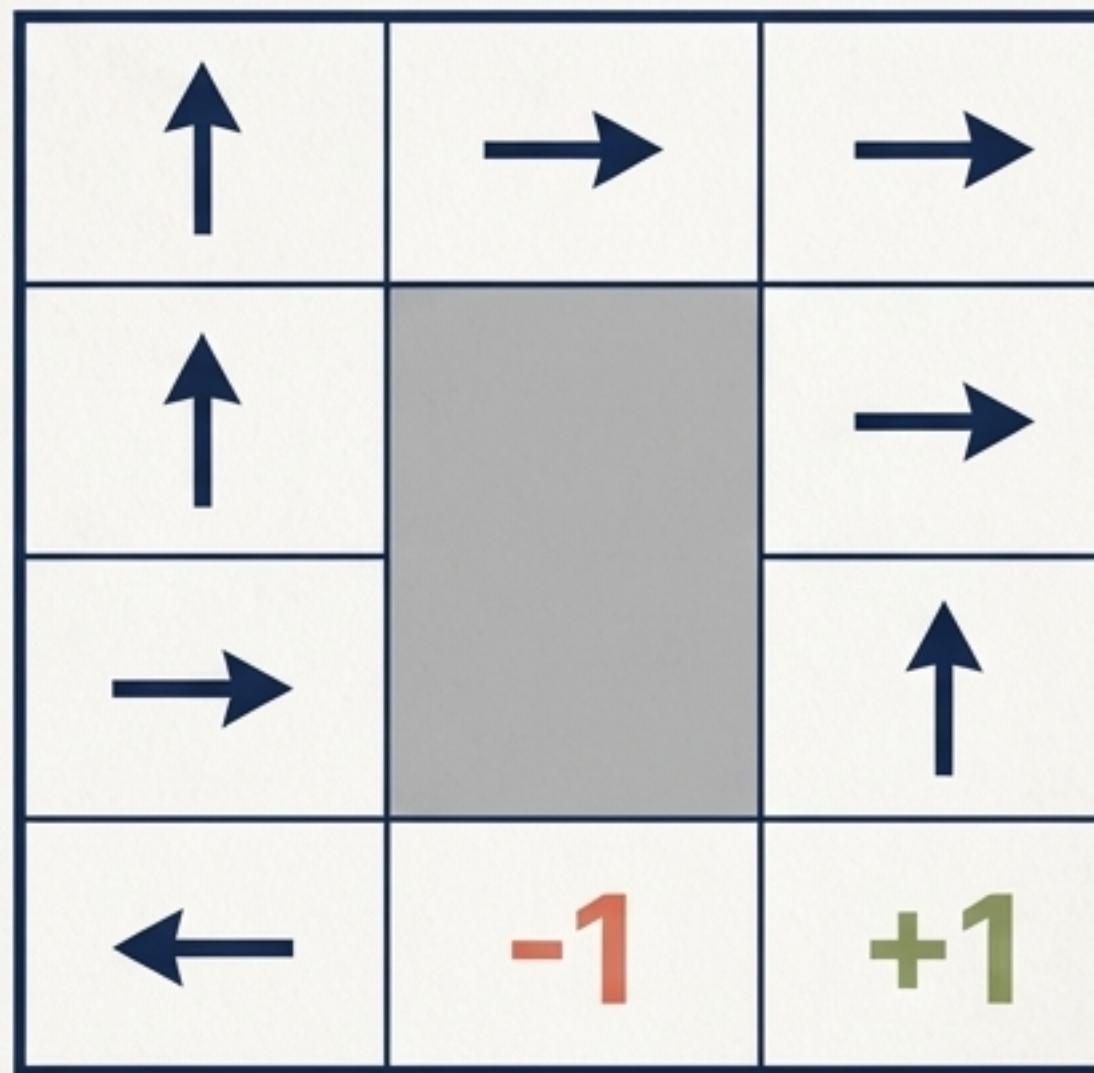
γ (gamma) là **hệ số chiết khấu (discount factor)**, ưu tiên các phần thưởng trước mắt hơn các phần thưởng trong tương lai xa.

Mục tiêu học: Tìm ra giá trị hữu dụng $U(s)$ cho mỗi trạng thái.

Bước đầu tiên: Đánh giá một chiến lược có sẵn (Học Thủ Động)

Kịch bản: Trong Học Thủ Động (Passive Learning), chính sách π của tác nhân là cố định. Tác nhân luôn thực hiện hành động $\pi(s)$ trong trạng thái s .

Mục tiêu: Chỉ đơn giản là học hàm hữu dụng $U_\pi(s)$ - tức là, tìm hiểu xem chính sách cố định này tốt đến mức nào.



(a) Chính sách cố định π (Fixed Policy π)

0.812	0.868	0.918
0.762		0.660
0.705		0.655
0.611	-1.000	+1.000

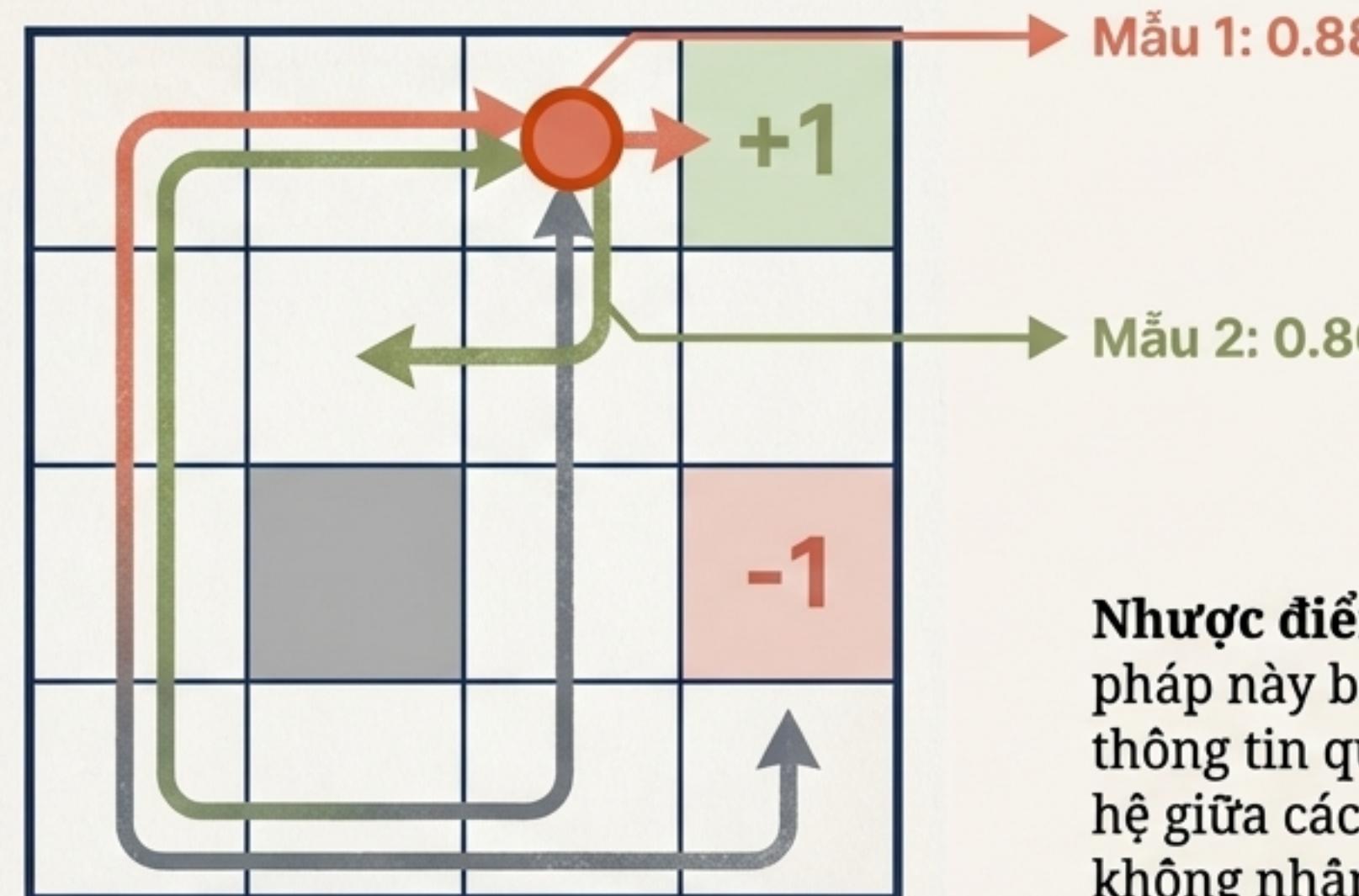
(b) Mục tiêu: Học các giá trị này (Goal: Learn these values)

Cách tiếp cận 1: Học từ các lần thử hoàn chỉnh (Ước tính Hữu dụng Trực tiếp)

Ý tưởng: Hữu dụng của một trạng thái là tổng phần thưởng kỳ vọng kể từ trạng thái đó trở đi ('reward-to-go'). Mỗi lần thử (trial) cung cấp một *mẫu* về giá trị này cho mỗi trạng thái đã ghé thăm.

Phương pháp:

1. Thực hiện nhiều lần thử từ đầu đến cuối.
2. Với mỗi trạng thái đã ghé thăm trong một lần thử, tính tổng phần thưởng nhận được từ đó cho đến cuối.
3. Giá trị hữu dụng của trạng thái đó được ước tính bằng cách lấy trung bình của tất cả các mẫu 'reward-to-go' đã quan sát.



$$U(1,3) \approx \text{trung bình}(0.88, 0.80, \dots)$$

Nhược điểm lớn: Phương pháp này bỏ qua một thông tin quan trọng: mối liên hệ giữa các trạng thái. Nó không nhận ra rằng hữu dụng của một trạng thái liên quan đến hữu dụng của các trạng thái kế tiếp của nó (theo phương trình Bellman). Do đó, nó hội tụ rất chậm.

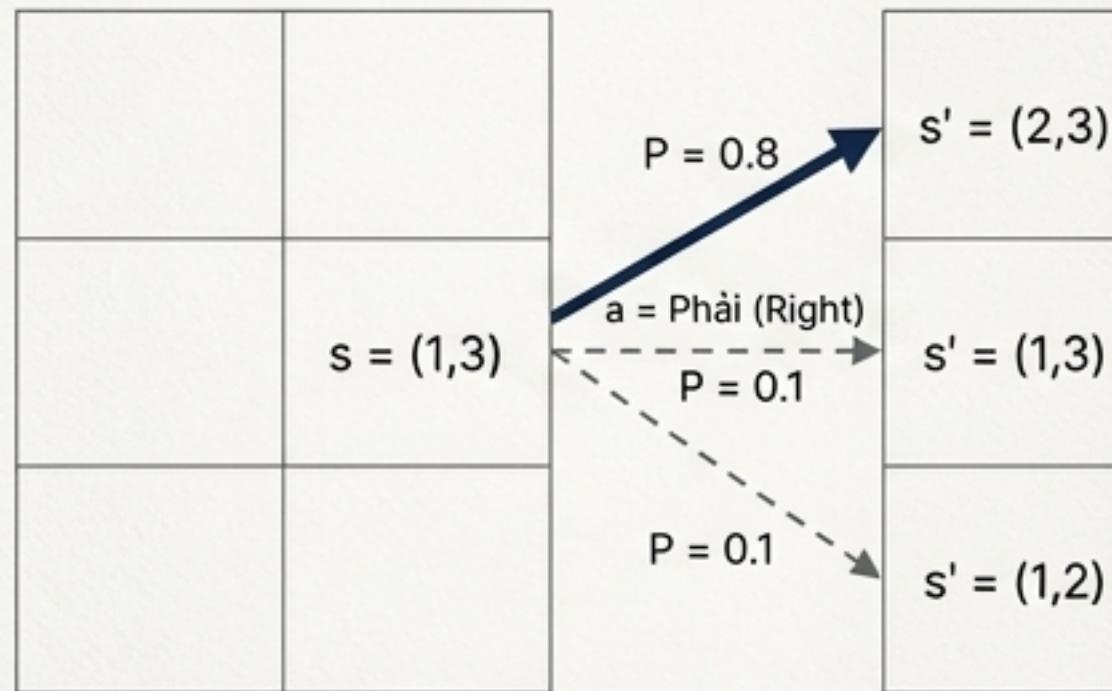
Cách tiếp cận 2: Xây dựng bản đồ và lập kế hoạch (Lập trình Động Thích ứng - ADP)

Ý tưởng: Tận dụng mối liên hệ giữa các trạng thái bằng cách học mô hình của môi trường, sau đó giải Quy trình Quyết định Markov (MDP) tương ứng.

Phương pháp (Dựa trên mô hình - Model-Based):

1. **Học mô hình chuyển đổi:** Ước tính $P(s' | s, a)$.
2. **Học hàm phần thưởng:** Quan sát $R(s)$.
3. **Giải phương trình Bellman:** $U_\pi(s) = R(s) + \gamma \sum P(s' | s, \pi(s))U_\pi(s')$

1. Học Mô hình (Learn the Model)



Ví dụ: "Từ $(1,3)$ đi Phải, 80% đến $(2,3)$ "

2. Giải tìm Hữu dụng (Solve for Utilities)

$U(1,1) = 0.611$	$U(1,2) = 0.705$	$U(1,3) = 0.762$
$U(2,1) = -1.000$	$U(2,2) = N/A$	$U(2,3) = 0.868$
$U(3,1) = +1.000$	$U(3,2) = 0.655$	$U(3,3) = 0.660$
$U(4,1) = 0.918$	$U(4,2) = 0.660$	$U(4,3) = 0.812$

Ưu điểm: Sử dụng hiệu quả thông tin, hội tụ nhanh hơn nhiều so với ước tính trực tiếp.

Nhược điểm: Không khả thi với không gian trạng thái lớn (ví dụ: cờ hậu có $\sim 10^{50}$ trạng thái).

Cách tiếp cận 3: Học từ mỗi bước đi (Học Khác biệt Thời gian - TD)

Ý tưởng (Phi mô hình - Model-Free):

Không cần học toàn bộ mô hình chuyển đổi.
Thay vào đó, sử dụng các chuyển đổi quan sát được để điều chỉnh trực tiếp các ước tính hữu dụng sao cho chúng nhất quán cục bộ.

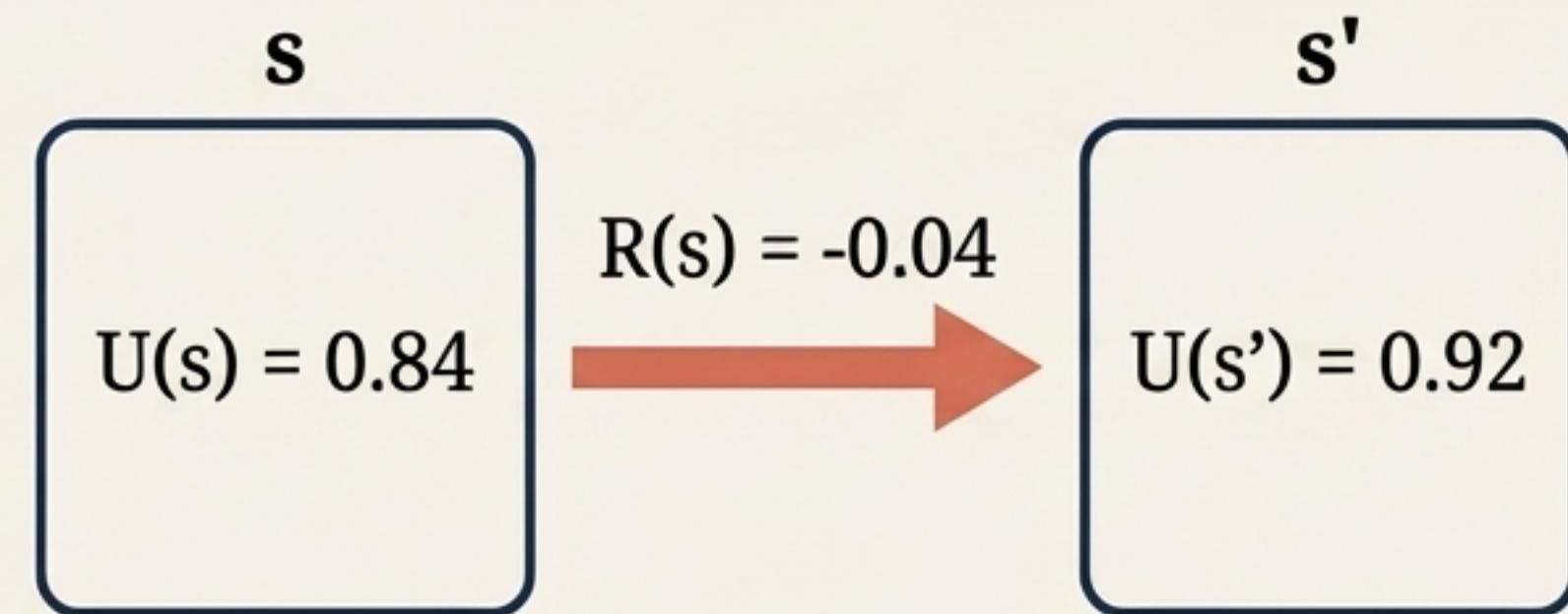
Quy tắc cập nhật TD:

$$U_{\pi}(s) \leftarrow U_{\pi}(s) + \alpha * (R(s) + \gamma U_{\pi}(s') - U_{\pi}(s))$$

Thành phần $(R(s) + \gamma U_{\pi}(s') - U_{\pi}(s))$ được gọi là **sai số khác biệt thời gian (temporal difference error)**.

So sánh TD và ADP:

- TD điều chỉnh một trạng thái để phù hợp với trạng thái kế tiếp *quan sát được*.
- ADP điều chỉnh một trạng thái để phù hợp với *tất cả* các trạng thái kế tiếp có thể xảy ra.



Cập nhật $U(s)$ (Updating $U(s)$)

$$U_{\pi}(s) \leftarrow 0.84 + \alpha * (-0.04 + \underbrace{\gamma * 0.92 - 0.84}_{\text{Sai số TD}})$$

$$\rightarrow 0.84 + \alpha * (0.04) \rightarrow 0.844 \text{ (với } \alpha=0.1, \gamma=1\text{)}$$

Từ bị động sang chủ động: Tác nhân phải tự quyết định

- **Thay đổi:** Tác nhân không còn có một chính sách cố định. Nó phải học *cái gì cần làm*. Đây là **Học Chủ Động (Active Learning)**.
- **Vấn đề mới:** Sau khi học một mô hình và tính toán hữu dụng, tác nhân có nên luôn chọn hành động được cho là tốt nhất không?
- **Dilemma cốt lõi:** **Khai thác (Exploitation)** vs. **Thăm dò (Exploration)**.



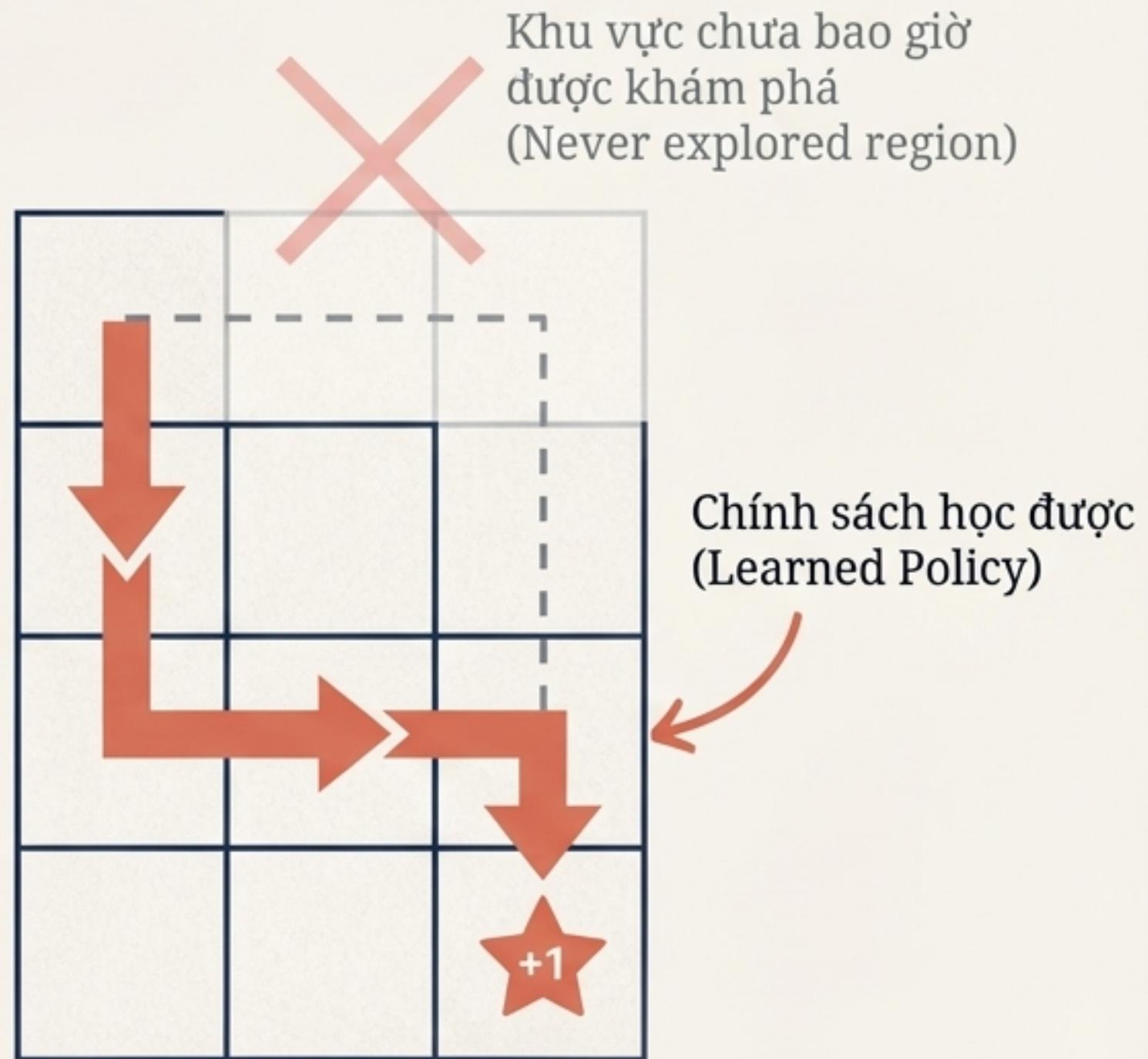
Cạm bẫy của sự tham lam

Tác nhân tham lam (Greedy Agent): Một tác nhân luôn thực hiện hành động mà nó tin là tối ưu dựa trên mô hình đã học hiện tại.

Thí nghiệm: Một tác nhân ADP tham lam trong thế giới 4x3.

- Ban đầu, nó khám phá.
- Trong lần thử thứ 39, nó tìm thấy một con đường đến phần thưởng +1 (con đường phía dưới).
- Sau đó, nó **mắc kẹt** vào chính sách này. Nó không bao giờ khám phá các trạng thái ở phía trên, và do đó không bao giờ tìm ra con đường tối ưu thực sự.

Kết luận: Việc chọn hành động tối ưu dựa trên một mô hình *chưa hoàn hảo* có thể dẫn đến kết quả dưới mức tối ưu.



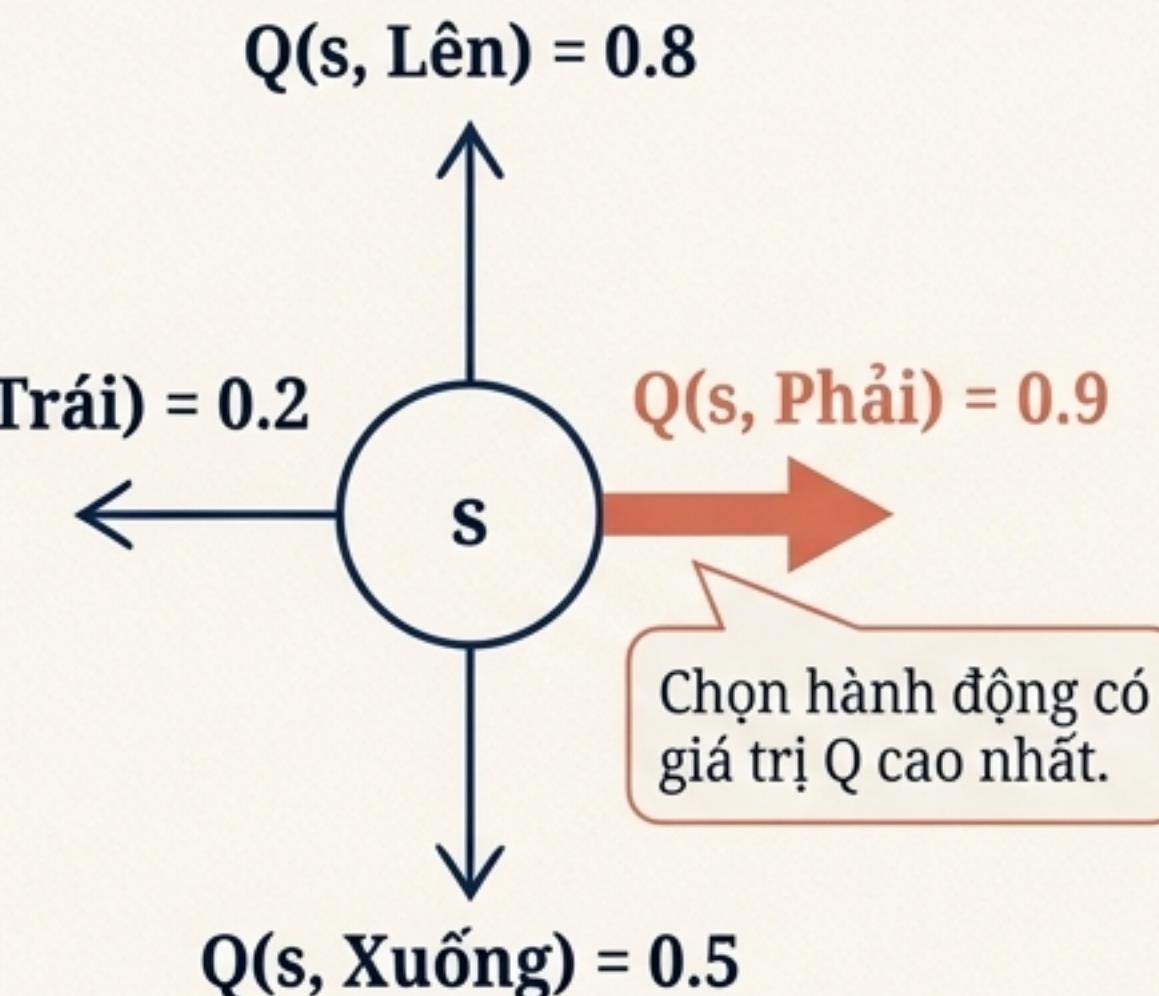
Q-Learning: Học giá trị của hành động

Vấn đề: Một tác nhân TD (phi mô hình) đã học $U(s)$ vẫn cần một mô hình để chọn hành động.

Giải pháp: Thay vì học $U(s)$, hãy học $Q(s, a)$ - hàm hữu dụng của hành động.

$Q(s, a)$ là giá trị kỳ vọng của việc thực hiện hành động a trong trạng thái s .

$$U(s) = \max_a Q(s, a)$$



Lợi ích lớn: Với các giá trị Q , tác nhân **không cần mô hình** để chọn hành động.

Đây là một thuật toán **phi mô hình (model-free)** và **ngoài chính sách (off-policy)**.

Quy tắc cập nhật TD cho Q-Learning:

$$Q(s, a) \leftarrow Q(s, a) + \alpha * (R(s) + \gamma * \max_{a'} Q(s', a') - Q(s, a))$$

Thực tế phũ phàng: Khi thế giới quá lớn

Vấn đề về quy mô: Các phương pháp trước đây giả định chúng ta có thể lưu trữ $U(s)$ hoặc $Q(s, a)$ trong một bảng.

Ví dụ:

- Backgammon (Cờ tào cáo): $\sim 10^{20}$ trạng thái.
- Chess (Cờ vua): $\sim 10^{40}$ trạng thái.
- Thậm chí một robot đơn giản với các cảm biến liên tục cũng có vô số trạng thái.

Chúng ta cần một cách để **tổng quát hóa (generalize)** kiến thức từ các trạng thái đã ghé thăm sang các trạng thái chưa từng thấy.



Làm sao để ước tính giá trị ở đây?
(How to estimate value here?)

Giải pháp: Học một hàm xấp xỉ, không phải một bảng

Ý tưởng: Sử dụng **Xấp xỉ Hàm (Function Approximation)** để biểu diễn hàm hữu dụng hoặc hàm Q.

Thay vì một bảng, chúng ta sử dụng một hàm có tham số, ví dụ như một mạng nơ-ron hoặc:

$$\hat{U}_\theta(s) = \theta_1 * f_1(s) + \theta_2 * f_2(s) + \dots + \theta_n * f_n(s)$$

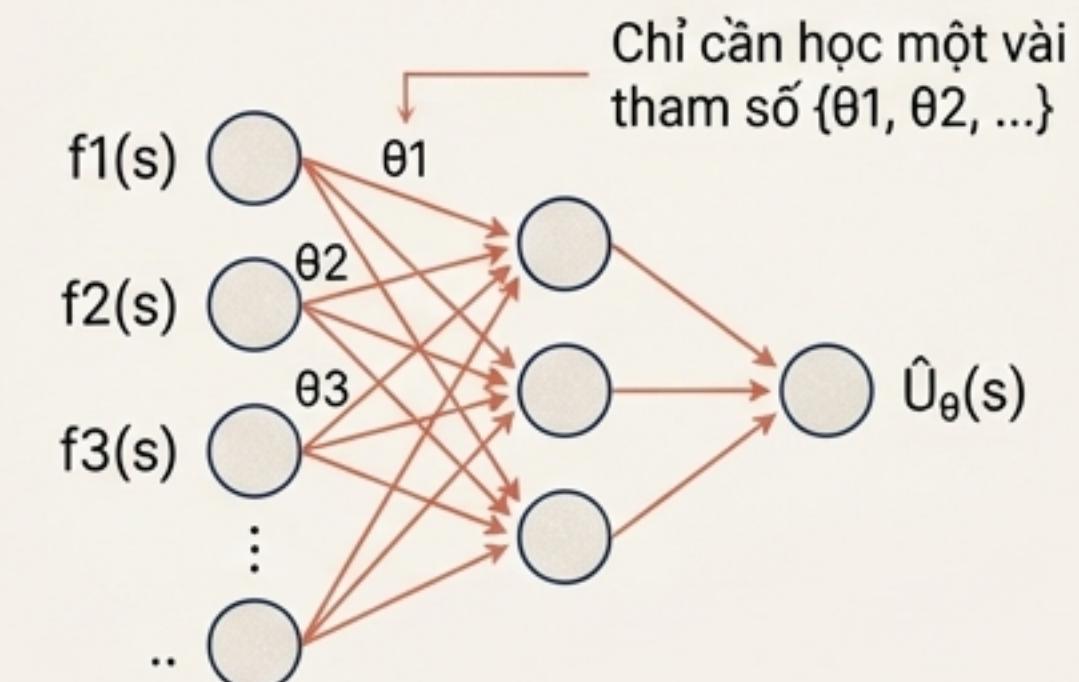
Bảng tra cứu (Tabular)

Trạng thái (State)	Giá trị (Value)
s1	s1 -> 0.812
s2	s2 -> 0.762
s3	s3 -> 0.901
s4	s4 -> 0.654
s5	s5 -> 0.888
...	...



Nén & Tổng quát hóa
(Compression & Generalization)

Xấp xỉ Hàm (Approximation)



$$\text{Học: } \theta_i \leftarrow \theta_i + \alpha * [\text{TD_error}] * (\partial \hat{U}_\theta(s) / \partial \theta_i)$$

Một con đường khác: Tìm kiếm trực tiếp chính sách tốt nhất

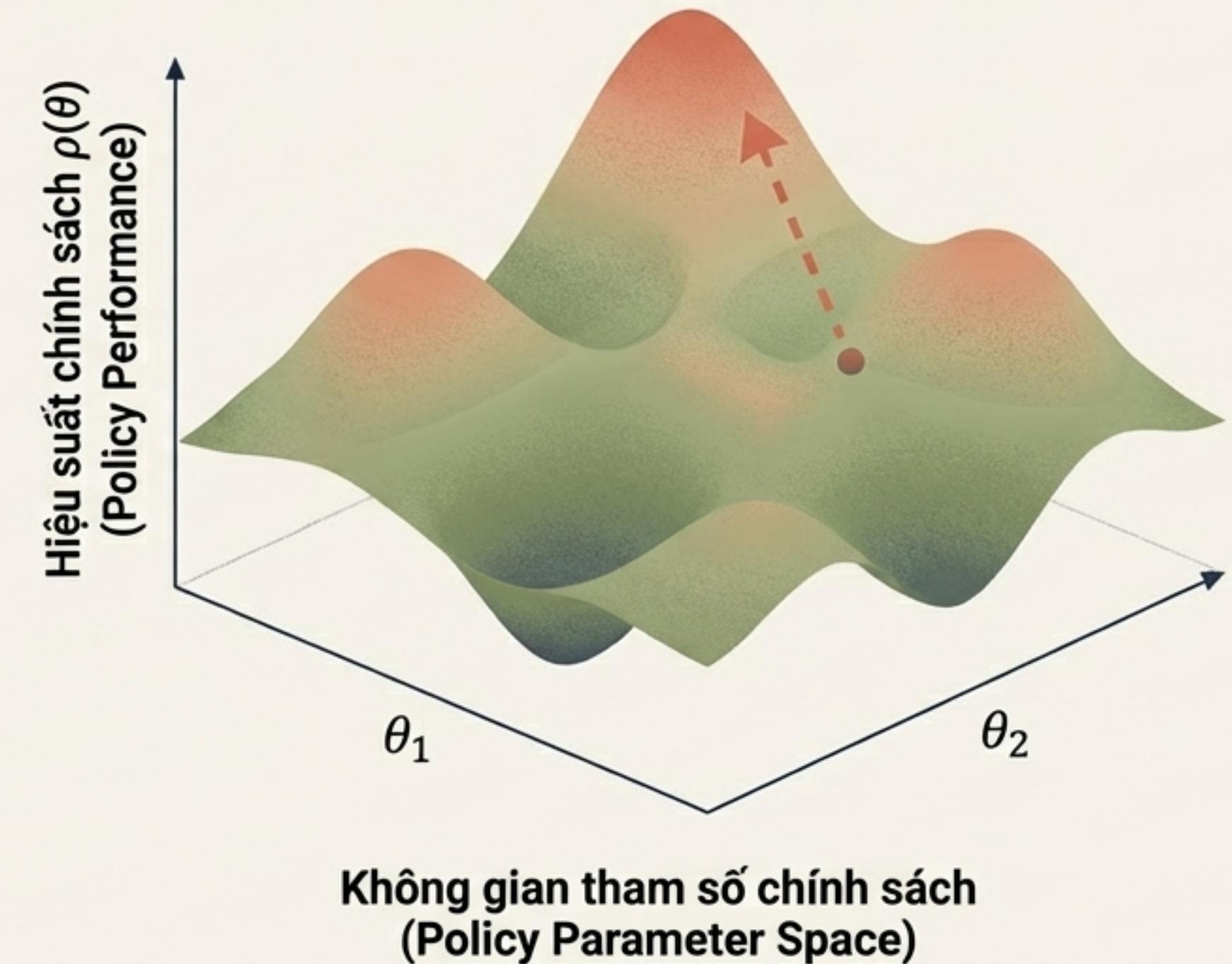
Ý tưởng (Tìm kiếm Chính sách - Policy Search):

- Thay vì học một hàm giá trị (U hoặc Q), tại sao không học trực tiếp một chính sách có tham số $\pi_\theta(s, a)$?
- $\pi_\theta(s, a)$ có thể là xác suất chọn hành động a trong trạng thái s .

Phương pháp:

1. Bắt đầu với một chính sách π_θ ngẫu nhiên.
2. Chạy thử và đo lường hiệu suất của nó.
3. Điều chỉnh các tham số θ theo hướng cải thiện.
4. Lặp lại.

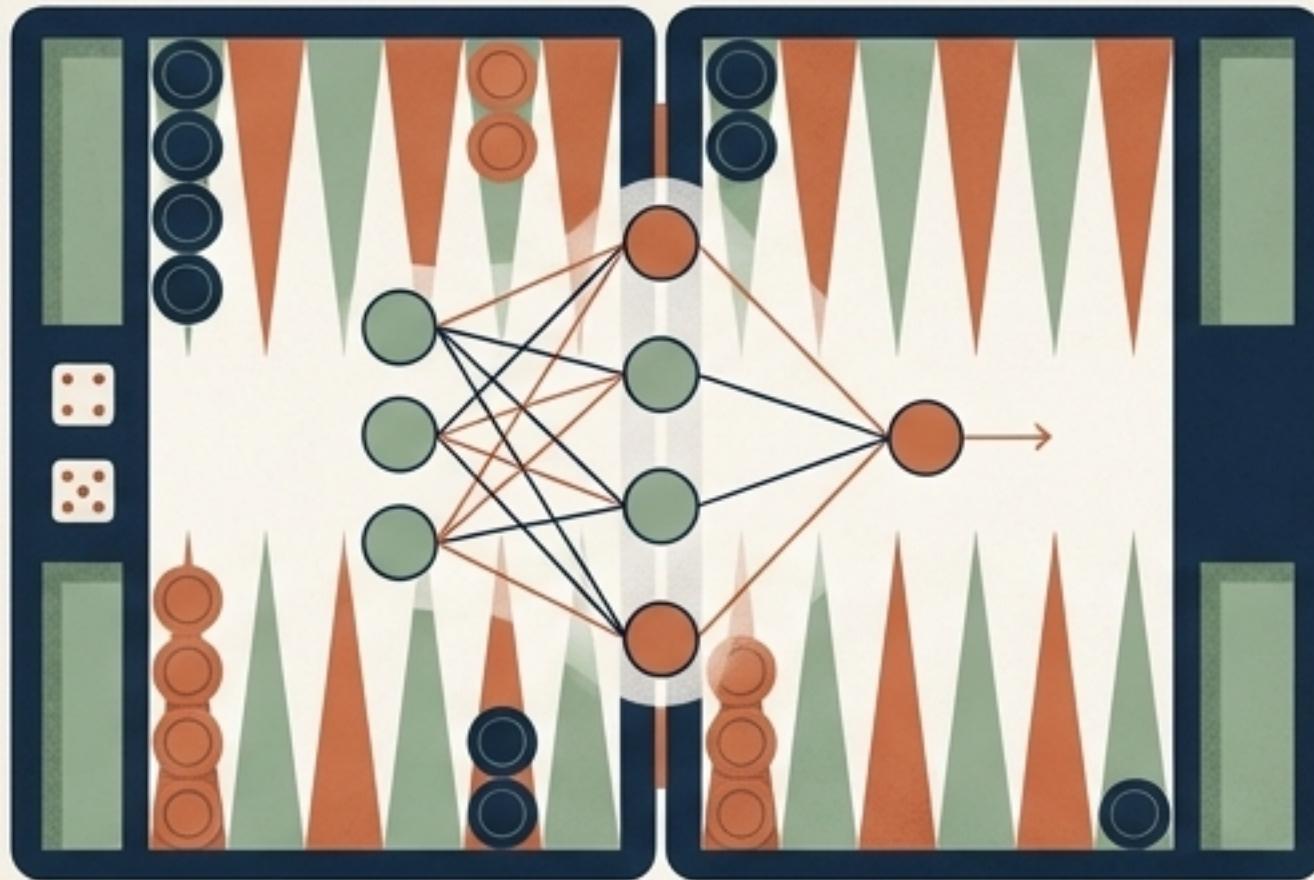
Khi nào nên sử dụng: Hữu ích trong không gian hành động liên tục hoặc khi biểu diễn chính sách đơn giản hơn biểu diễn hàm giá trị.



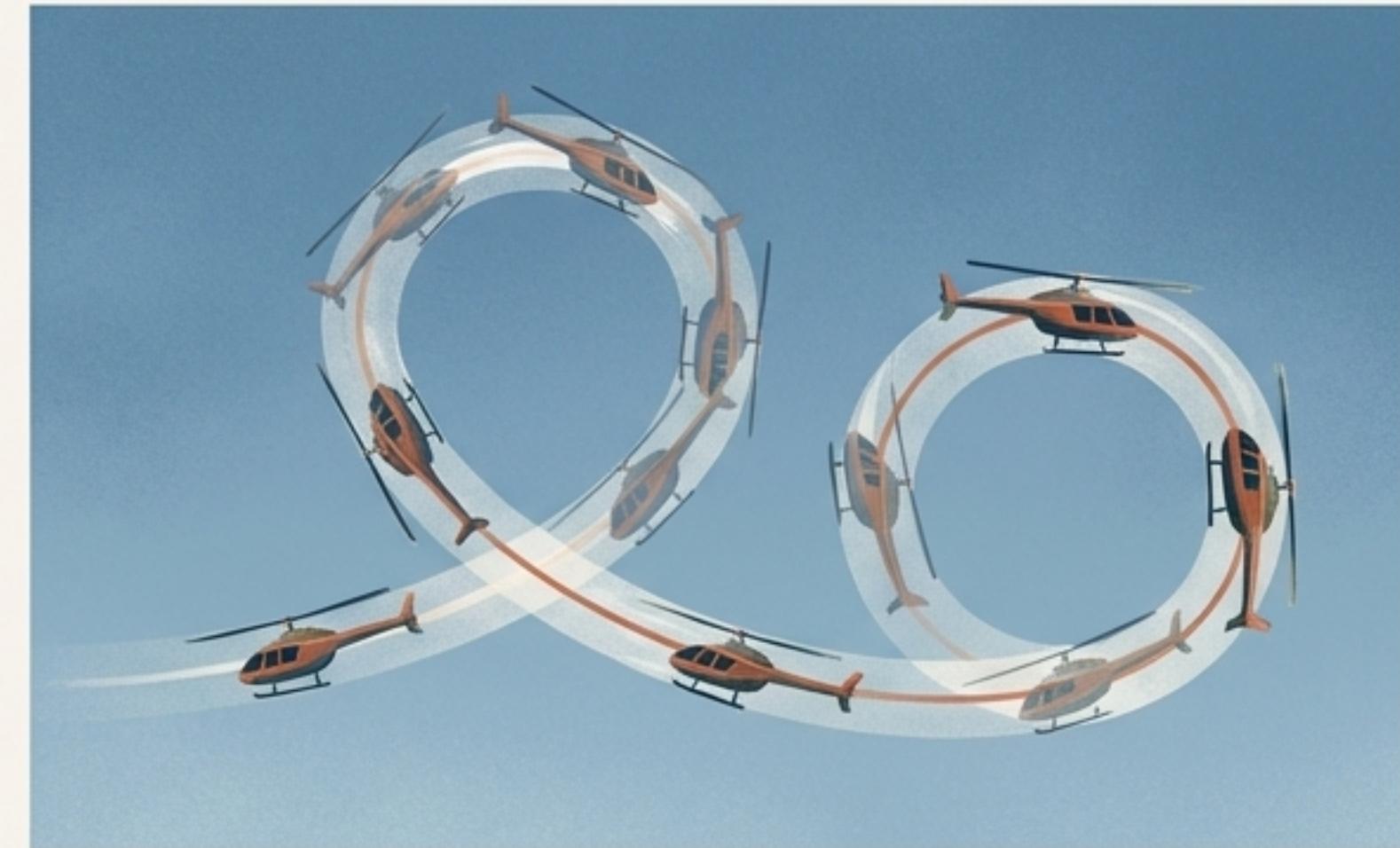
Không gian tham số chính sách
(Policy Parameter Space)

Sức mạnh phi thường của Học Tăng Cường

TD-Gammon (1992)



Điều khiển trực thăng tự hành (2004)



- Sử dụng học khác biệt thời gian (TD) và một mạng nơ-ron.
- Học thông qua việc tự chơi hàng trăm nghìn ván cờ.
- Tín hiệu phần thưởng duy nhất là thắng/thua ở cuối trận.
- Đạt đến trình độ thi đấu ngang hàng với những người chơi giỏi nhất thế giới.

Kết luận: Học Tăng Cường cho phép các tác nhân đạt được hiệu suất siêu phàm trong các miền phức tạp, thường chỉ bằng cách học từ kinh nghiệm thô, giống như con người.

- Sử dụng các thuật toán Tìm kiếm Chính sách (Policy Search).
- Học cách thực hiện các thao tác nhào lộn phức tạp.
- Vượt xa khả năng của một phi công chuyên gia.