



Định nghĩa Markov Decision Process


- Một **MDP** là một khuôn khổ toán học dùng để mô hình hóa các bài toán ra quyết định trong đó tác nhân (agent) tương tác với môi trường (environment) qua nhiều bước thời gian rời rạc (discrete time steps).
 - Tại mỗi bước thời gian, tác nhân quan sát trạng thái hiện tại của môi trường, chọn một hành động, và nhận đồng thời phần thưởng (reward) lẫn trạng thái tiếp theo.
 - Điều quan trọng: **tính chất Markov** ở đây có nghĩa là, xác suất để đi từ trạng thái hiện tại sang trạng thái kế tiếp chỉ phụ thuộc vào trạng thái hiện tại và hành động đã chọn, không phụ thuộc vào lịch sử các trạng thái/hành động trước đó. 
-

3. Ví dụ minh họa: Con thỏ lựa chọn rau củ

3.1 Tình huống ngay lập tức (Bandit Rabbit)

- **Thuyết minh:**
 - Ban đầu, nếu con thỏ ("bandit rabbit") chỉ quan tâm đến phần thưởng tức thời, nó sẽ chọn luôn **cà rốt (carrot)** ở bên phải vì thưởng +10 lớn hơn +3 của súp lơ (broccoli) bên trái.
 - Trường hợp này tương tự bài toán K-Armed Bandit, vì chỉ xét reward hiện tại và không lường trước hậu quả sau đó. 
- **Giải thích:**
 - Nếu môi trường chỉ là "thấy cà rốt bên phải ăn được +10, thấy súp lơ bên trái ăn được +3" và không tính yếu tố nào khác, thì con thỏ hoàn toàn sẽ "khai thác" (exploit) chọn cà rốt mà không cân nhắc thêm.

3.2 Tình huống có nhiều hoàn cảnh (Contextual Rabbit)

- **Thuyết minh:**
 - Khi con thỏ gặp hoàn cảnh khác (ví dụ: cà rốt nằm bên trái, súp lơ nằm bên phải), nó sẽ chọn **cà rốt** ở bên trái vì vẫn là reward +10 lớn hơn +3.
 - Như vậy, khác với bài bandit, giờ đây "hành động đúng" phụ thuộc vào "tình huống" (situation) hiện tại—đó chính là khái niệm trạng thái (state). 
- **Giải thích:**
 - Bài toán Bandit không mô hình hóa được khái niệm "tình huống" đang thay đổi liên tục—nó chỉ biết có một tập các cần gạt (arms) cố định.
 - MDP mở rộng thêm khái niệm "trạng thái" giúp tác nhân lựa chọn hành động phù hợp với từng tình huống.

3.3 Tình huống dài hạn (Long-Term Rabbit)

- **Thuyết minh:**

- Giả sử phía sau cà rốt bên phải là một con hổ. Nếu con thỏ chọn “đi phải” để ăn cà rốt, nó sẽ lấy được +10 bình thường, nhưng ngay sau đó có nguy cơ bị hổ ăn thịt → tổng kết quả dài hạn có thể thành $-\infty$ (nếu bị hổ).
 - Trong khi đó, nếu đi trái chỉ ăn được +3, nhưng con thỏ sẽ an toàn và có thể sống sót về lâu dài.
 - Như vậy, nếu chỉ xét reward hiện tại (cà rốt +10), con thỏ sẽ phải trả giá rất đắt. Nhưng xét **quá trình dài hạn**, chọn “súp lơ” tuy ít reward ngay, nhưng đảm bảo an toàn tổng lợi ích tốt hơn. 📄
 - **Giải thích:**
 - Bài toán Bandit chỉ quan tâm reward tức thời, không mô hình tác động dài hạn.
 - MDP xét đến hệ quả của mỗi hành động về lâu dài, cho phép “đánh đổi” reward hiện tại và tương lai dựa vào xác suất và chế độ chuyển trạng thái.
-

4. Các thành phần cơ bản của MDP

1. Agent (Tác nhân):

- Là thực thể đang được huấn luyện để ra quyết định.
- Ví dụ: con thỏ quyết định ăn gì; robot quyết định di chuyển như thế nào. 📄

2. Environment (Môi trường):

- Là phần còn lại của thế giới mà tác nhân tương tác.
- Environment quản lý trạng thái bên trong, nhận hành động từ agent và trả về phần thưởng lẫn trạng thái mới. 📄

3. State (Trạng thái, S):

- Mô tả “hoàn cảnh” hiện tại của agent trong môi trường.
- Ở ví dụ con thỏ: trạng thái có thể là “cà rốt bên phải, súp lơ bên trái” hay “súp lơ bên phải, cà rốt bên trái”, hoặc “có hổ phía sau cà rốt”.
- Ở ví dụ robot: trạng thái có thể là “robot đang đứng ở ô (3,5) trong bản đồ lưới”. 📄

4. Action (Hành động, A):

- Là lựa chọn mà agent thực hiện ở một trạng thái.
- Ví dụ con thỏ: ăn cà rốt (go right) hay ăn súp lơ (go left).
- Ví dụ robot: di chuyển lên, xuống, trái, phải hoặc ở nguyên vị trí. 📄

5. Policy (π):

- Là “chiến lược” hay “quy tắc” xác định cách agent chọn hành động dựa trên trạng thái hiện tại.
- Một chính sách có thể là deterministic (tại mỗi trạng thái luôn chọn hành động nhất định) hoặc stochastic (phân phối xác suất cho từng hành động).

- Ví dụ: policy của con thỏ có thể là “nếu phát hiện hố phía sau cà rốt, luôn chọn súp lơ; ngược lại chọn cà rốt”. 📄

6. Reward (Phần thưởng, R):

- Là tín hiệu vô hướng (scalar) môi trường trả về cho agent ngay sau khi agent thực hiện hành động.
- Reward có thể dương (thưởng) hoặc âm (phạt).
- Ví dụ: +10 nếu con thỏ ăn được cà rốt; +3 nếu ăn được súp lơ; -100 nếu gặp hố (có thể coi là phạt nặng). 📄

7. Transition Dynamics (P):

- Được gọi là hàm xác suất chuyển tiếp:

$$P(s', r | s, a) = \Pr(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a).$$

- Nó cho biết, khi agent ở trạng thái s và thực hiện hành động a , xác suất để chuyển sang trạng thái s' và nhận được reward r .
- Bởi vì P là phân phối xác suất trên tất cả cặp (s', r) , nên


$$\sum_{s', r} P(s', r | s, a) = 1 \quad \text{với mọi } (s, a).$$

- Hàm này nhấn mạnh tính **Markov** (Markov property): quyết định tương lai chỉ dựa vào tình hình hiện tại, không phụ thuộc lịch sử. 📄

5. Ví dụ trong bài toán robot điều hướng

• Mô tả bối cảnh:

- Agent: một robot tự động di chuyển trong một môi trường dạng “grid world”.
- Environment: lưới có các ô (cells), trong đó có các chướng ngại, có điểm khởi hành và đích đến.
- State: vị trí hiện tại của robot trong lưới (ví dụ ô (x, y)).
- Action: di chuyển lên, xuống, trái, phải hoặc dừng lại.
- Policy: luật quyết định “nên đi hướng nào tại mỗi ô” để tiến gần đến đích, tránh chướng ngại.
- Reward:
 - +1 khi robot chạm đến ô đích;
 - 0 cho mỗi bước di chuyển bình thường;
 - -1 (hoặc $-\infty$) nếu đụng chướng ngại hoặc ra khỏi lưới.
- Transition Dynamics:
 - Ví dụ: khi ở ô (i, j) và chọn “di chuyển lên”, với xác suất 0.8 robot lên được ô $(i-1, j)$, với xác suất 0.1 robot vô tình bị gió thổi sang trái (ô $(i, j-1)$), và 0.1 sang phải (ô $(i, j+1)$).

Nếu ô đó là chương ngại, robot vẫn ở nguyên vị trí hoặc nhận phạt. 

6. Tính chất “Markov” trong MDP


- **Markov Property:**

- Nếu biết trạng thái hiện tại S_t , hành động A_t , thì thông tin về các trạng thái/hành động trước đó không còn giúp dự đoán tốt hơn về trạng thái kế tiếp S_{t+1} hay reward R_{t+1} .
- Tức:

$$\Pr(S_{t+1}, R_{t+1} \mid S_t, A_t, S_{t-1}, A_{t-1}, \dots) = \Pr(S_{t+1}, R_{t+1} \mid S_t, A_t).$$


- Nhờ tính Markov, ta chỉ cần quan tâm đến $P(s', r \mid s, a)$ mà không phải giữ lịch sử dài. 

- **Ý nghĩa thực tiễn:**


- Khi thiết kế hoặc ước lượng MDP, chỉ cần nắm “trình trạng hiện tại” (state) và “kinh nghiệm quá khứ” được tích lũy dưới dạng giá trị hàm hoặc policy, chứ không phải giữ lại toàn bộ lịch sử.
 - Giúp đơn giản hóa mô hình và tính toán khi giải bài toán RL. 
-

7. Tóm tắt (Summary)

1. MDP là gì?


- Khuôn khổ mô hình hóa bài toán ra quyết định có trạng thái và hành động lặp đi lặp lại.
- Bao gồm tập trạng thái S , tập hành động A , hàm chuyển tiếp xác suất $P(s', r \mid s, a)$, và reward. 

2. Các thành phần chính:

- Agent: tác nhân ra quyết định.
- Environment: thế giới xung quanh mà agent tương tác.
- State: mô tả hoàn cảnh hiện tại.
- Action: lựa chọn của agent.
- Policy: chiến lược chọn action dựa trên state.
- Reward: tín hiệu phản hồi dạng vô hướng.
- Transition Dynamics: hàm xác suất xác định state và reward kế tiếp. 

3. Ví dụ con thỏ:

- Cho thấy sự khác biệt giữa bài Bandit (chỉ quan tâm reward tức thời) và MDP (cân nhắc ngữ cảnh và hậu quả dài hạn).

- Trong MDP, lựa chọn phải dựa trên thông tin hiện tại (ví dụ có hoặc không có hổ), không chỉ reward ngay trước mắt. 

4. Động lực của MDP:

- Khi thực hiện hành động a ở trạng thái s , môi trường trả về một bộ (s', r) theo xác suất $P(s', r | s, a)$.
- Tính chất Markov cho phép chúng ta dự đoán chỉ từ (s, a) mà không phụ thuộc lịch sử.



Gợi ý áp dụng và mở rộng:

- Trong thực tế, để giải bài toán MDP, người ta định nghĩa thêm các khái niệm như hàm giá trị state $V^\pi(s)$, hàm giá trị action $Q^\pi(s, a)$ theo policy π , và sử dụng các thuật toán như Value Iteration, Policy Iteration, Q-Learning, SARSA, v.v.
- Nếu môi trường không rõ trước hàm $P(s', r | s, a)$, ta có thể dùng model-free methods (RL không sử dụng mô hình), ví dụ Q-Learning.