

2. Khi nào mô hình trở nên không chính xác?

1. Mô hình không đầy đủ (Incomplete model)

- Agent chưa từng thử tất cả các hành động ở hầu hết các trạng thái, nên nhiều transition vẫn **thiếu** trong model 📄 .

2. Môi trường thay đổi

- Ngay cả những transition đã lưu trước đó cũng có thể **không còn khớp** với thực tế khi môi trường thay đổi (ví dụ reward hay trạng thái kế tiếp khác đi) 📄 .

3. Ảnh hưởng của mô hình không chính xác khi planning


- **Sai lệch (Inaccurate model)**: Những transition giả lập từ model có thể **khác** so với kết quả thực.
- Nếu agent dùng những transition **sai** này để cập nhật giá trị (value) hay chính sách (policy), thì **cập nhật có thể đi lệch hướng**, làm policy thực tế tồi đi so với khi chỉ học từ dữ liệu thật 📄 .
- Vì vậy agent phải duy trì model luôn **chính xác**—nghĩa là cần tiếp tục **khám phá** để cập nhật những phần model còn thiếu hoặc lỗi thời.

4. Ví dụ minh họa: Thỏ và Cà rốt

- Ban đầu model cho rằng, nếu thỏ ở ô trái cùng và chọn “di chuyển phải”, thì sẽ **đến ô chứa cà rốt**.
- Thực tế, khi thực hiện, thỏ chỉ đến **ô giữa** (cà rốt vẫn xa).
- Sau khi trải nghiệm, thỏ **cập nhật model** để biết “từ ô trái cùng, action ‘phải’ dẫn đến ô giữa” thay vì ô đích 📄 .
- Qua ví dụ này, ta thấy planning với model cũ sẽ đưa ra policy sai, và agent cần **tương tác lại** để sửa model.

5. Trade-off khám phá–khai thác mới

- Khi model không chính xác, agent có hai lựa chọn:
 1. **Khai thác** model hiện có để tìm policy tốt nhất theo giả định model đúng → có thể tệ khi model sai.


2. Khám phá môi trường thật để **cập nhật model**, rồi mới tiếp tục planning.
- Đây là một **exploration-exploitation trade-off** bậc hai, bên cạnh trade-off trong chọn action .
-

6. Giải pháp: Khuyến khích thăm lại (Bonus rewards for exploration)

- Để agent **tự động** ưu tiên khám phá những (s, a) lâu chưa được cập nhật model, ta thêm **bonus** vào reward khi planning:

$$r^{\text{plan}} = r + \kappa \sqrt{\tau},$$

trong đó

- r là reward gốc từ model,
 - τ là **thời gian** (số step) kể từ lần thăm cuối cùng (s, a) trong môi trường thật,
 - κ là hằng số nhỏ điều khiển mức ảnh hưởng của bonus.
- Lưu ý: τ **không** được cập nhật trong planning loop (vì đó không phải real visit) .
-

7. Thuật toán Dyna-Q+

Kết hợp bonus trên vào bước planning của Dyna-Q, ta được **Dyna-Q+**:

1. Sau mỗi real-step


- Cập nhật $Q(s,a)$ như Q-Learning, lưu model $M[s,a] = (r,s')$, và cập nhật time-stamp last-visit của (s,a) .

2. Trong mỗi planning step (lặp k lần)

- Chọn ngẫu nhiên (\tilde{s}, \tilde{a}) từ model M .
- Lấy $(\tilde{r}, \tilde{s}') = M[\tilde{s}, \tilde{a}]$, tính $\text{bonus} = \kappa \sqrt{\tau(\tilde{s}, \tilde{a})}$.
- Cập nhật:

$$Q(\tilde{s}, \tilde{a}) \leftarrow Q(\tilde{s}, \tilde{a}) + \alpha [(\tilde{r} + \kappa \sqrt{\tau}) + \gamma \max_{a'} Q(\tilde{s}', a') - Q(\tilde{s}, \tilde{a})].$$

8. So sánh Dyna-Q và Dyna-Q+

Tiêu chí	Dyna-Q	Dyna-Q+
Khuyến khích khám phá	Dùng fixed exploration bonus (không thay đổi)	Bonus tăng dần theo $\sqrt{\tau}$, ưu tiên những (s,a) lâu chưa thăm 
Chiến lược planning	Cập nhật tất cả (s,a) trong model, không phân biệt	Ưu tiên (s,a) có τ cao, tức còn model sai nhiều hơn
Reward bổ sung	Không thêm reward ngoài reward gốc	Thêm reward dựa trên độ "cũ" của (s,a), tăng khả năng sửa model
Hiệu quả vs model sai	Có thể lãng phí planning vào transitions đã chính xác	Tập trung sửa những phần model có sai lệch cao

9. Tóm tắt (Summary)

- Mô hình không chính xác xảy ra do model thiếu dữ liệu hoặc môi trường thay đổi.
- **Planning với model sai** có thể làm policy tệ đi, nên cần trade-off khám phá–khai thác phụ để cập nhật lại model.
- **Dyna-Q+** giải quyết vấn đề này bằng **bonus reward** tỉ lệ với $\sqrt{\tau}$, tự động ưu tiên planning về những state-action lâu không được thăm, từ đó sửa nhanh model cũ