2. Kiến trúc Dyna (The Dyna Architecture)

Dyna (Sutton, 1990) là một **khuôn khổ lai** ("hybrid") trong RL, tích hợp đồng thời:

- **Model-free learning** (ví dụ Q-Learning, SARSA) – học trực tiếp từ trải nghiệm thực (real experience).
- **Model-based planning** – dùng một **mô hình** (model) giả lập môi trường để "học thêm" từ **data nhân tạo** (simulated experience).

2.1. Các thành phần chính

1. **Environment & Policy**
   - Agent tương tác với môi trường thật, chọn hành động theo chính sách hiện tại, nhận về các transitions (s, a, r, s').
2. **Model Learning**
   - Từ mỗi transition thực, ta học (hoặc cập nhật) **mô hình** của môi trường:
     - Ghi nhận $P(s' \mid s, a)$ và $R(s, a)$ (trong Tabular Dyna-Q thường là lưu "bản ghi" cuối cùng).
     - Kết quả là ta có thể "kể lại" (replay) transition đó bao nhiêu lần tùy thích.
3. **Model (Sample/Simulated Experience)**
   - Khi planning, ta **sample** (ngẫu nhiên hoặc controlled) một (s, a) đã từng gặp, dùng model để sinh ra (s', r) giả lập.
   - Sử dụng transition giả lập này để **cập nhật** hàm giá trị (Q hoặc V) y hệt như với dữ liệu thật.
4. **Search Control**
   - Quy định **chiến lược** lấy mẫu từ model:
     - *Random-sample one-step*: chọn ngẫu nhiên (s, a) trong lịch sử.
     - Hoặc ưu tiên (s, a) có độ bất định cao, hoặc gần terminal, v.v.
5. **Model-free Updates**
   - Dùng tất cả transitions—cả thật và giả lập—để update Q (hoặc V) theo công thức Q-Learning, SARSA, TD(0),...
6. **Experience Replay (Tùy chọn)**
   - Lưu trữ cả transitions thật và giả lập trong buffer, rút ngẫu nhiên để giảm tương quan giữa các cập nhật và tăng tính ổn định.

---

3. Thuật toán Tabular Dyna-Q

Đây là phiên bản đơn giản nhất, dùng Q-Learning làm "core" và thêm planning k bước mỗi lần:

plaintext Sao chép Chỉnh sửa

```
Input: learning rate α, discount γ, exploration ε, planning steps k Initialize Q(s,a) tùy ý (ví dụ 0) cho mọi (s,a) Khởi tạo model M rỗng for each episode do Khởi tạo state s ← s₀ while s không phải terminal do • Chọn action a theo ε-greedy dựa trên Q(s, ·) • Thực thi a, quan sát r, next state s' • Cập nhật Q-Learning: Q(s,a) ← Q(s,a) + α [r + γ max_{a'} Q(s',a') − Q(s,a)] • Cập nhật model M: M[s,a] ← (r, s') • Gán s ← s' // **Planning**: lặp k lần for i = 1 to k do • Chọn ngẫu nhiên một cặp (s̃, ã) từ tập keys(M) • Lấy (r̃, s̃') = M[s̃, ã] • Cập nhật Q-Learning giả lập: Q(s̃,ã) ← Q(s̃,ã) + α [r̃ + γ max_{a'} Q(s̃',a') − Q(s̃,ã)] end for end while end for
```

- **Mỗi bước tương tác thật**: vừa update Q, vừa ghi model.
- **Mỗi lần planning**: dùng model replay k transitions đã lưu, thu thêm dữ liệu để update Q.

Kết quả: so với Q-Learning thuần túy, Dyna-Q thường hội tụ nhanh hơn (với cùng số real-step) vì tận dụng thêm k cập nhật "miễn phí" từ model.

---

4. Ví dụ minh họa

Giả sử agent đang học trong một **Gridworld** đơn giản:

- Mỗi ô là một state, agent có 4 action.
- Khi đi vào bẫy (hole) nhận $r = -1$ và episode kết thúc; khi đến goal nhận $r = +1$. Các bước khác $r = 0$.

4.1. Tương tác thật

1. Agent ở ô A, chọn action "Right" → di chuyển đến ô B, $r = 0$.
2. Cập nhật Q(A,"Right") và ghi model: M[A,"Right"] = (0, B).

4.2. Planning (k=5)

- Lúc này model lưu được một số cặp (s, a). Ví dụ keys(M) = {(A,"Right"), (B,"Down"), …}.
- Chọn ngẫu nhiên (B,"Down"), giả lập: (r̃=0, s̃'=C).
- Cập nhật Q(B,"Down") như thể mới gặp transition thật.
- Lặp thêm 4 lần với các cặp khác.

Kết quả, Q sẽ được lan truyền thông tin về reward từ goal ngược về các state xa hơn mà không cần agent phải thật sự đi đến goal mỗi lần.

---

5. So sánh Dyna vs Q-Learning

Tiêu chí	Q-Learning thuần túy	Dyna-Q (với k planning)
Sample Efficiency	Cần nhiều real-step để lan truyền reward xa	Với mỗi real-step được k updates từ model, nhanh hơn
Công thức update	Chỉ dùng (s, a, r, s') thật	Dùng cả thật và giả lập
Khám phá & ổn định	Dễ dao động khi real data ít	Ổn định hơn nhờ replay và planning
Phức tạp triển khai	Đơn giản	Cần thêm phần lưu và sample model
Yêu cầu	Không cần model	Cần học hoặc cung cấp chính xác model

---

6. Ưu điểm và Khi nào dùng Dyna

- **Ưu điểm**
  - *Cải thiện hiệu quả mẫu* (sample efficiency): lan truyền reward nhanh.
  - *Linh hoạt*: kết hợp song song real và simulated experience.
  - *Có thể tận dụng experience replay* để tăng ổn định.
- **Khi nào nên dùng**
  - Môi trường tương tác thật **đắt tiền** (robot, sim-to-real).
  - Có thể **ước lượng mô hình** tương đối chính xác.
  - Muốn tận dụng cả **planning** (trong bộ nhớ) lẫn **learning** từ data thật.

Kết luận

Kiến trúc Dyna-Q là một trong những ví dụ tiêu biểu cho cách **kết hợp nhuần nhuyễn** giữa model-based planning và model-free learning, giúp agent học nhanh hơn và hiệu quả hơn. Tabular Dyna-Q với vài dòng code đơn giản đã thể hiện rõ sức mạnh của việc tái sử dụng transitions qua planning để tăng tốc độ hội tụ so với Q-Learning thuần túy.