8. Integrating Learning and planning

8.1 Recall

Value function from experience

MC , TD(λ)

prediction / control

8.2 learn a Model

MDP Markov Decision Process

$\langle S, A, P, R \rangle$

P: prob transition

R: Reward

$M = \langle P_\eta, R_\eta \rangle$ η: 参数.

$S_{t+1} \sim P_\eta(S_{t+1} \mid S_t, A_t)$ 概率性

$R = R_\eta(R_{t+1} \mid S_t, A_t)$ 决定性

Grid world

Model learning

$S_1, A_1 \longrightarrow R_2, S_2$

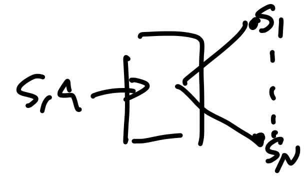
$S_2, A_2 \longrightarrow R_3, S_3$

$\vdots \qquad \vdots$

$S_{T-1}\ A_{T-1} \longrightarrow R_T\ S_T$

$s, a \xrightarrow{R_\eta} r$: regression problem

$s, a \xrightarrow{P_\eta} s'$: density function



SGD $\rightarrow$ η^*.

8.3 Table Lookup Model

$$\hat{P}^a_{s,s'} = \frac{1}{N(s,a)} \sum_{t=1}^{T} \mathbf{1}(S_t, A_t, \boxed{S_{t+1}} = s, a, \boxed{s'})$$

$$\hat{R}^a_s = \frac{1}{N(s,a)} \sum_{t=1}^{T} \mathbf{1}(S_t, A_t = s, a) R_t$$

More Sample from Model

$Q(s,a) \longrightarrow$ control

$S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1} \ldots\ldots$

planning by Model: if Model inaccurate, Serious bias

8.4 plan with a Model

$$M_\eta = \langle P_\eta, R_\eta \rangle$$

Solve: MDP.

$\langle S, A, P, R \rangle$

Use MDP to: Value iteration

Policy iteration

CoD: Curse of Dimensionality.

8.5 Sample-base. planning.

$S_{t+1} \sim P(S_{t+1} | S_t, A_t)$

$R_{t+1} = R_\eta(R_{t+1} | S_t, A_t)$

$S_1, A_1, R_2, S_2, A_2 \ldots$

Sarsa, MC, TD(λ) Q-learning

Random-sample one-step tabular Q-planning

Loop forever:

1. Select a state, $S \in \mathcal{S}$, and an action, $A \in \mathcal{A}(S)$, at random
2. Send S, A to a sample model, and obtain
   a sample next reward, R, and a sample next state, S'
3. Apply one-step tabular Q-learning to S, A, R, S':
   $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$

If Model inaccurate:

(1) Model-free RL

(2) Model uncertainty

(3) integrating experience and model

8.6. Integrating

Exp

① Model free → Q

② Train Model.

③ Using Model to plan → Q.

Tabular Dyna-Q

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Loop forever:
(a) $S \leftarrow$ current (nonterminal) state
(b) $A \leftarrow \varepsilon$-greedy(S, Q)
(c) Take action A; observe resultant reward, R, and state, S'
(d) $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
(e) $Model(S,A) \leftarrow R, S'$ (assuming deterministic environment)
(f) Loop repeat n times:
$S \leftarrow$ random previously observed state
$A \leftarrow$ random action previously taken in S
$R, S' \leftarrow Model(S,A)$
$Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

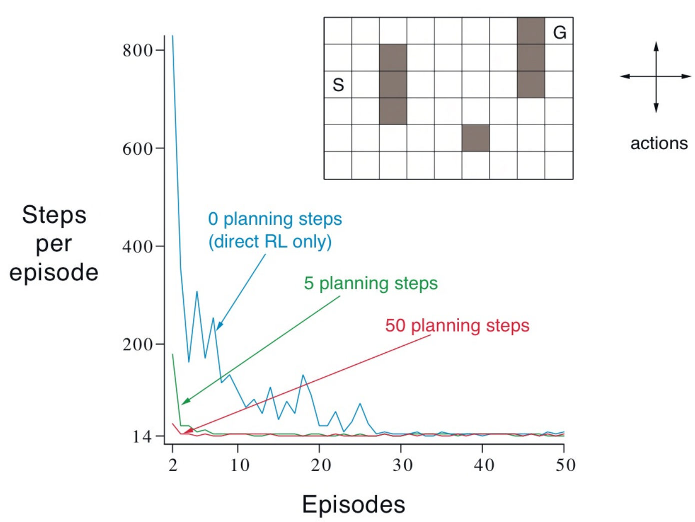


Figure 8.2: A simple maze (inset) and the average learning curves for Dyna-Q agents varying in their number of planning steps (n) per real step. The task is to travel from S to G as quickly as possible.

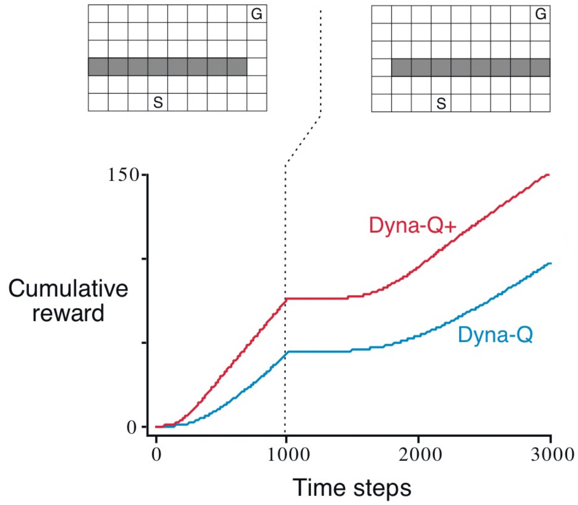


Figure 8.4: Average performance of Dyna agents on a blocking task. The left environment was used for the first 1000 steps, the right environment for the rest. Dyna-Q+ is Dyna-Q with an exploration bonus that encourages exploration. ■

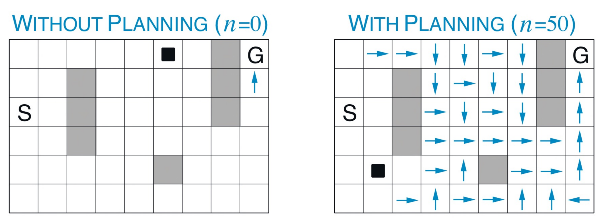


Figure 8.3: Policies found by planning and nonplanning Dyna-Q agents halfway through the second episode. The arrows indicate the greedy action in each state; if no arrow is shown for a state, then all of its action values were equal. The black square indicates the location of the agent. ■

8.7 Prioritize Sweeping for a deterministic Env.

8.7. Prioritized sweeping for a deterministic env

Prioritized sweeping for a deterministic environment

Initialize $Q(s,a)$, $Model(s,a)$, for all s,a, and $PQueue$ to empty
Loop forever:
  (a) $S \leftarrow$ current (nonterminal) state
  (b) $A \leftarrow policy(S, Q)$
  (c) Take action A; observe resultant reward, R, and state, S'
  (d) $Model(S,A) \leftarrow R, S'$ 추가(d)
  (e) $P \leftarrow |R + \gamma \max_a Q(S',a) - Q(S,A)|$.
  (f) if $P > \theta$, then insert S, A into $PQueue$ with priority P
  (g) Loop repeat n times, while $PQueue$ is not empty:
    $S, A \leftarrow first(PQueue)$
    $R, S' \leftarrow Model(S,A)$
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
    Loop for all $\bar{S}, \bar{A}$ predicted to lead to S:
      $\bar{R} \leftarrow$ predicted reward for $\bar{S}, \bar{A}, S$
      $P \leftarrow |\bar{R} + \gamma \max_a Q(S,a) - Q(\bar{S},\bar{A})|$.
      if $P > \theta$ then insert $\bar{S}, \bar{A}$ into $PQueue$ with priority P