RL: Lecture 18

Harvey Mudd College

April 6, 2020

Neil Rhodes

What is planning?

Tabular Dyna-Q Algorithm

Initialize $Q(s,a)$ and $Model(s,a)$ for all a, s

Loop forever:

1. $S \leftarrow$ current (nonterminal) state
2. $A \leftarrow \epsilon - \text{greedy}(S,Q)$
3. Take action A; observe resultant reward R and state S'
4. $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \text{max}_a Q(S',a) - Q(S,A)]$
5. $\text{Model}(S,A) \leftarrow (R, S')$ (assumes deterministic envirnoment)
6. Loop repeat n times

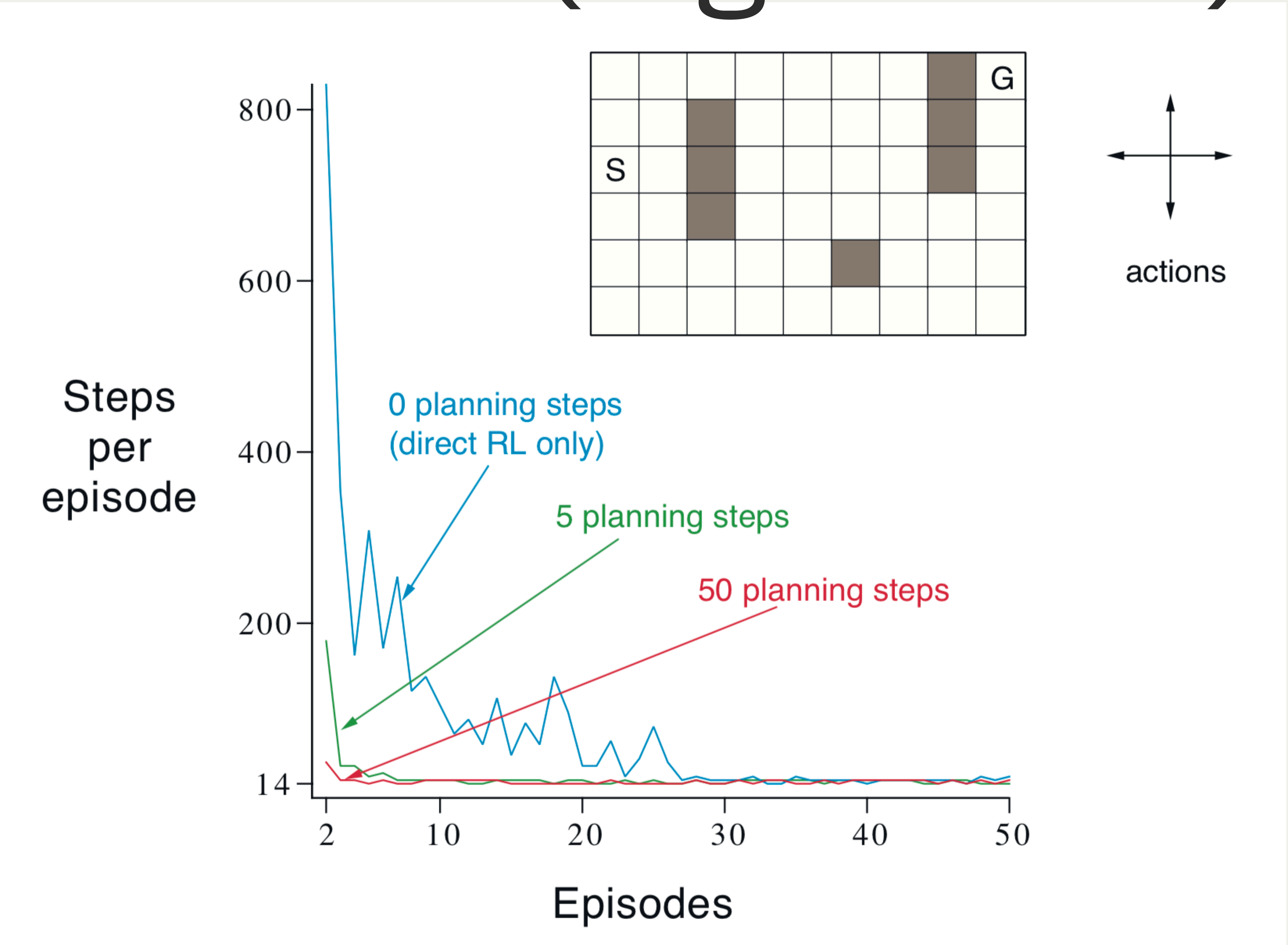
   $S \leftarrow$ random previously observed state

   $A \leftarrow$ random action previously taken in S

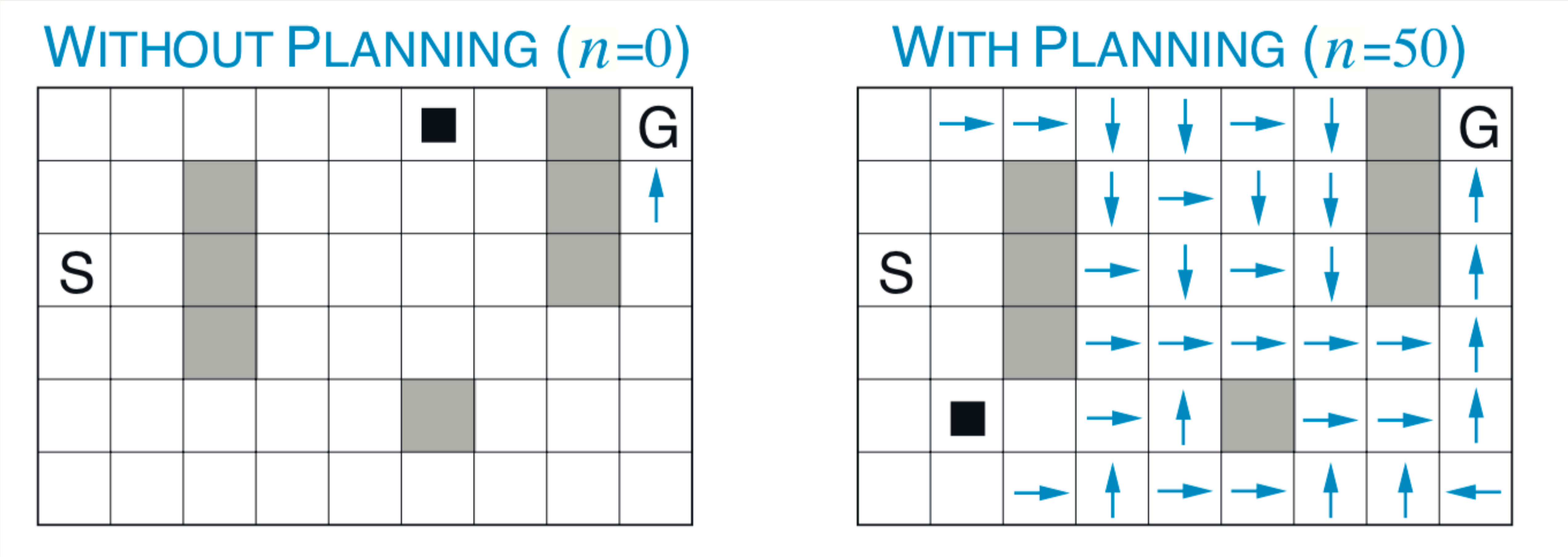
   $R, S' \leftarrow \text{Model}(S,A)$

   $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \text{max}_a Q(S',a) - Q(S,A)]$

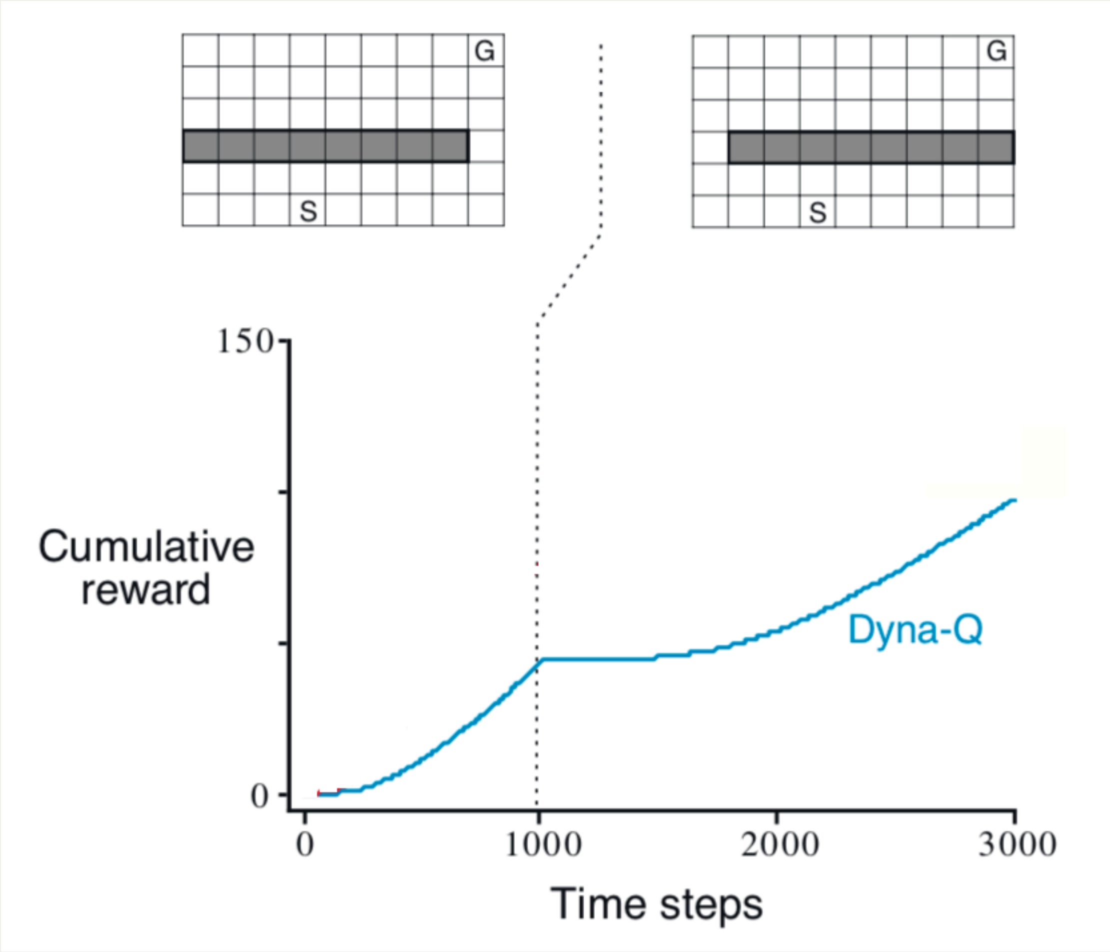
Dyna Maze (Figure 8.2)



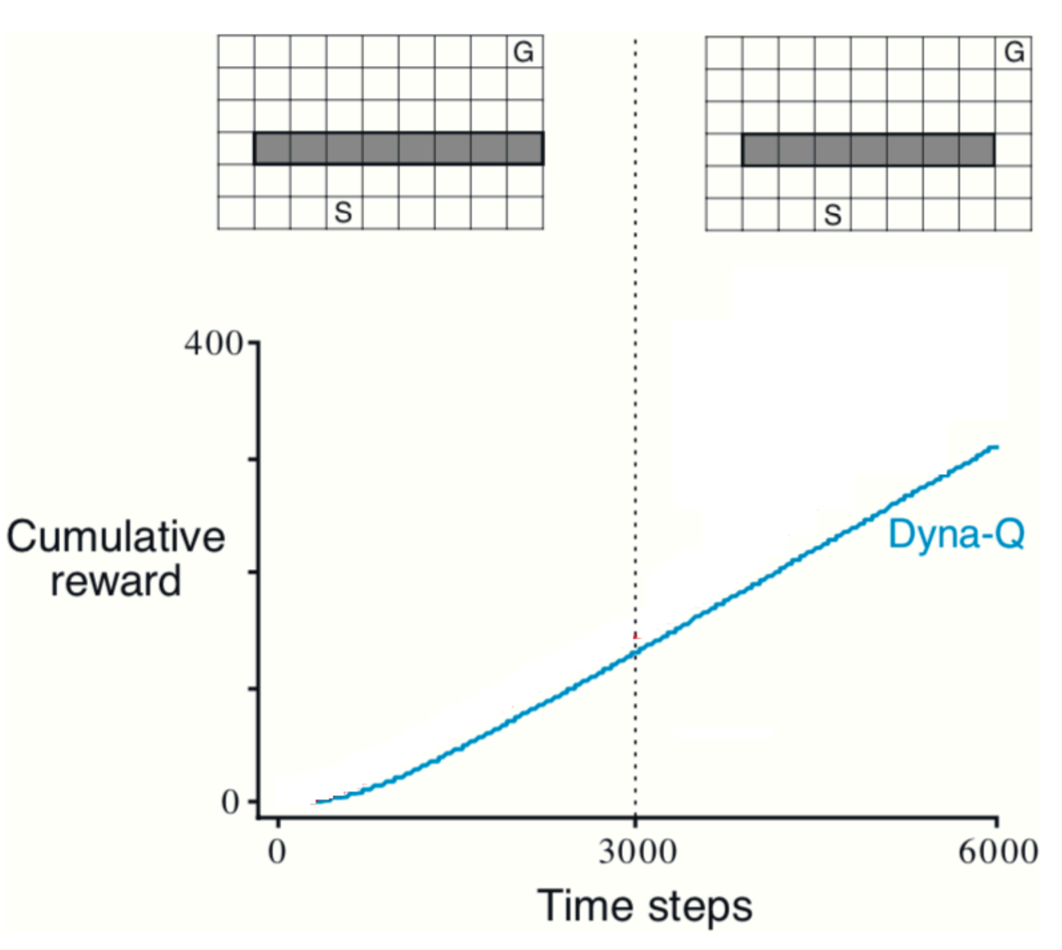
Dyna Maze (Figure 8.3)



When the Model is Wrong: Optimistic Model



When the Model is Wrong: Pessimistic Model



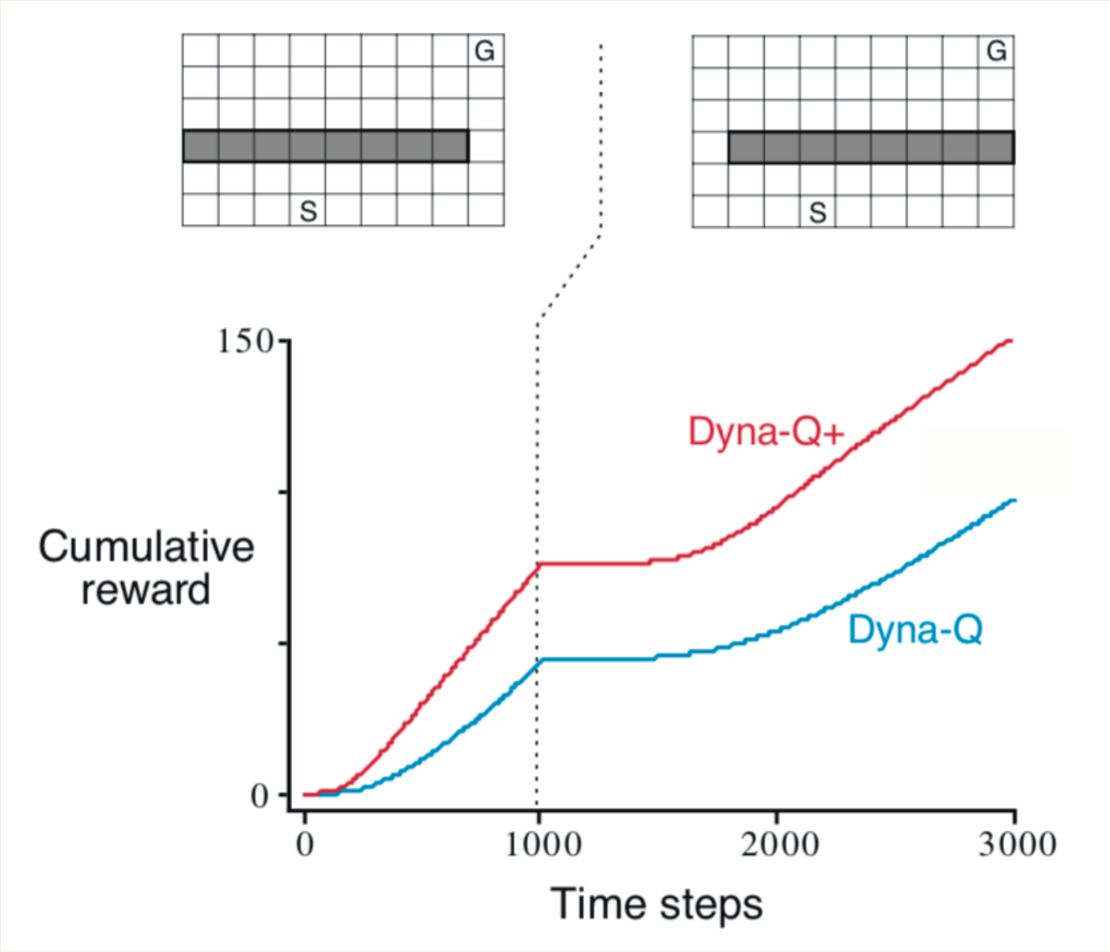
Dyna-Q+: Dyna-Q + heuristics for encouraging model updates

- Provide an implicit reward to exploring stale transitions

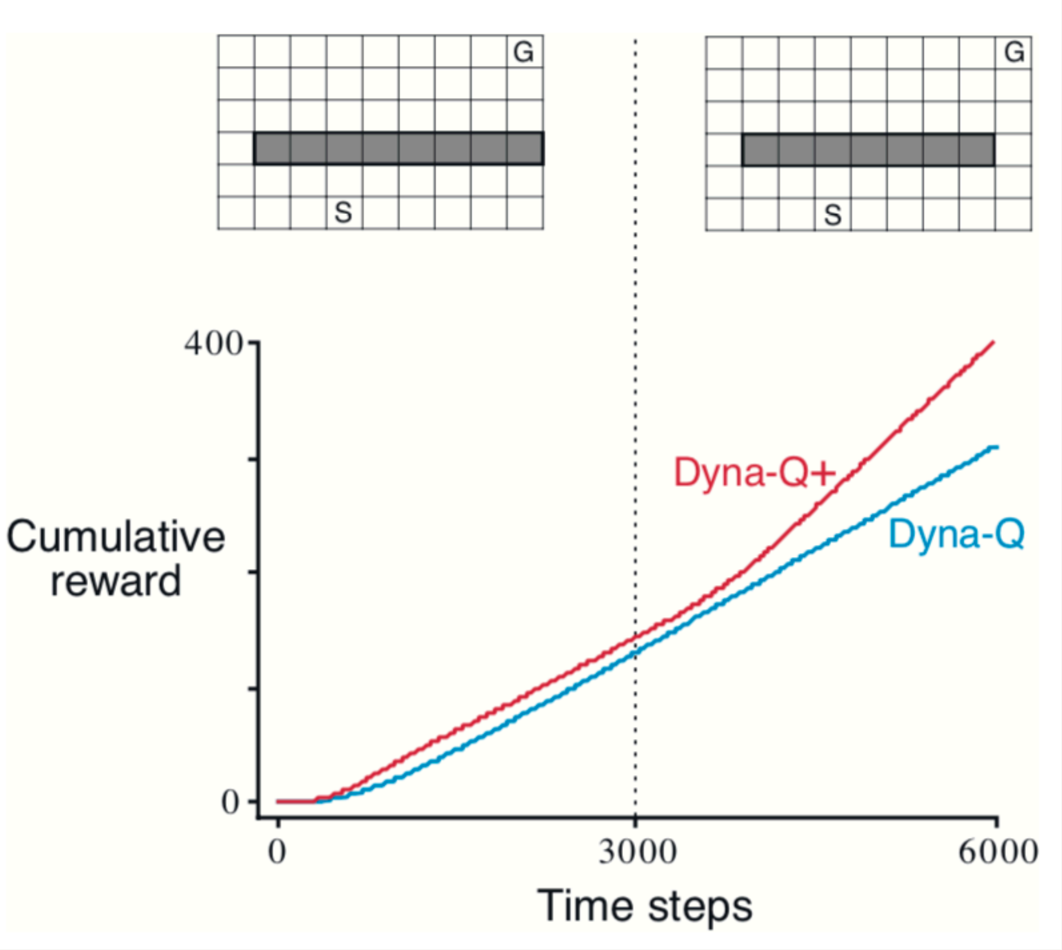
$$Q(S,A) \leftarrow Q(S,A) + \alpha[R + \kappa\sqrt{\tau(S,A)} + \gamma \mathbf{max}_a Q(S',a) - Q(S,A)]$$

- Allows actions that have never been tried from a state to be considered in planning (initial model: such an action leads back to the same state with a reward of 0)

When the Model is Wrong: Optimistic Model

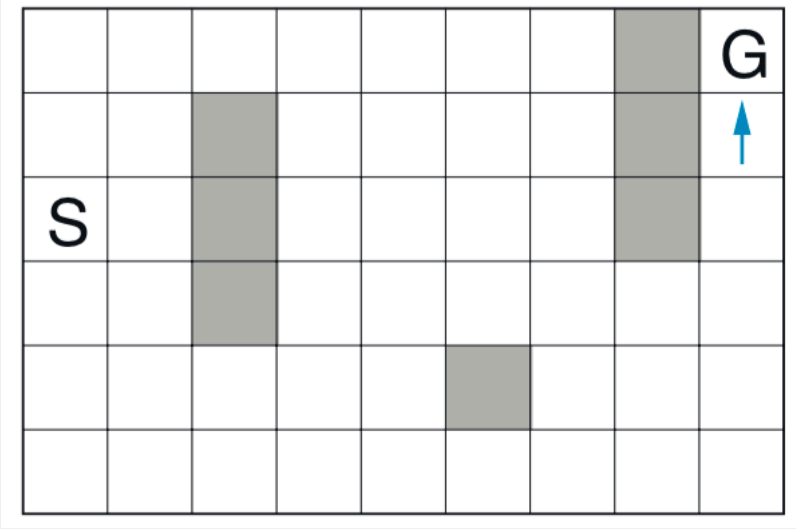


When the Model is Wrong: Pessimistic Model



Dyna-Q tries all state-action pairs uniformly.

Is there a better way?

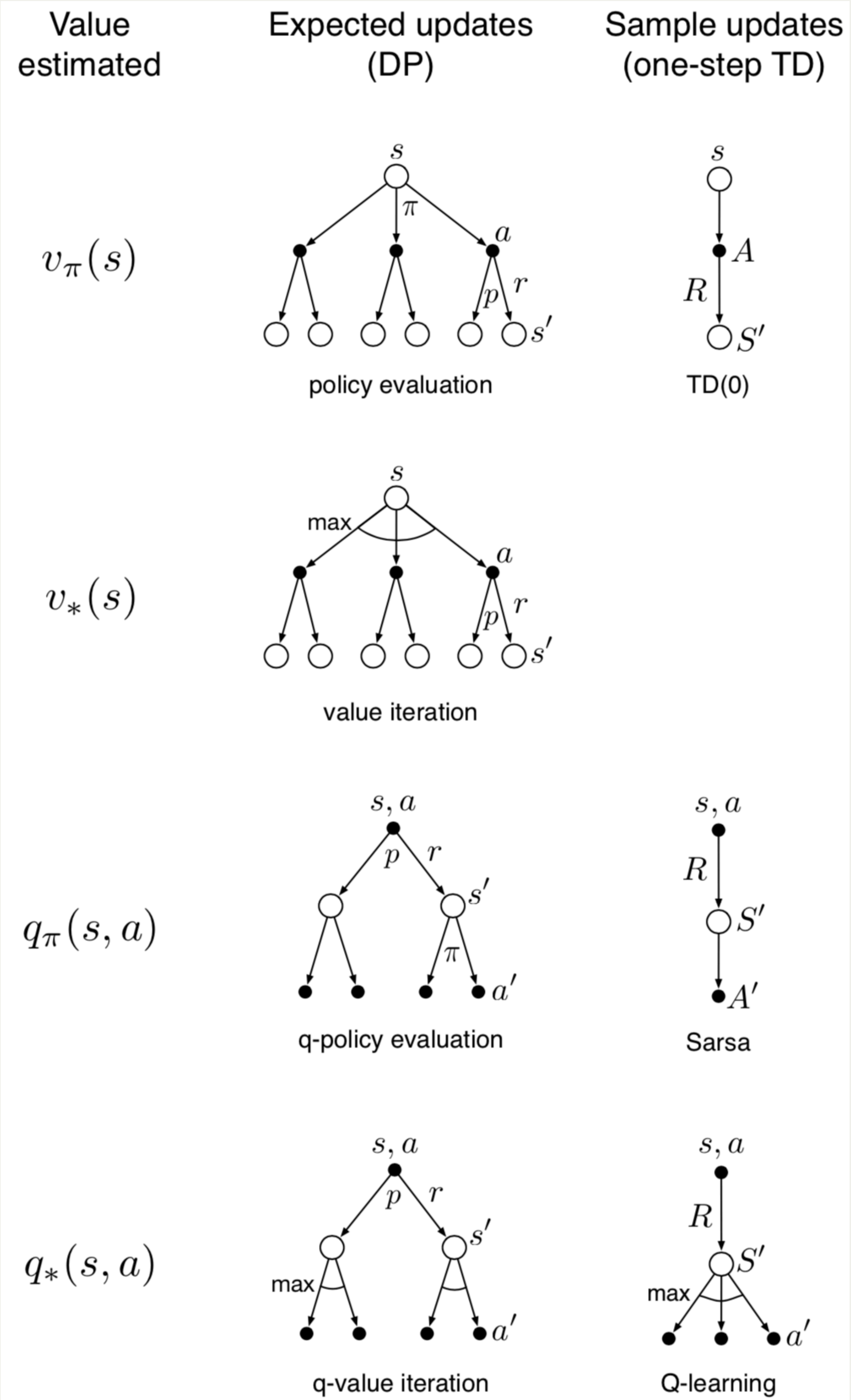


Prioritized Sweeping (Det. Env.)

Initialize $Q(s, a)$ and $Model(s, a)$ for all a, s, $PQueue$ to empty
Loop forever:
1. $S \leftarrow$ current (nonterminal) state; 2. $A \leftarrow \text{policy}(S, Q)$
3. Take action A; observe resultant reward R and state S'
4. $\text{Model}(S, A) \leftarrow (R, S')$ (assumes deterministic envirnoment)
5. $P = R + \gamma \text{max}_a Q(S', a) - Q(S, A)$
6. If $P > \Theta$, insert (S, A) into $PQueue$ with priority P
7. Loop repeat n times while $PQueue$ is not empty
   a. $S, A = first(PQueue); R, S' \leftarrow \text{Model}(S, A)$
   b. $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \text{max}_a Q(S', a) - Q(S, A)]$
   c. Loop for all $\bar{S}, \bar{A}$ predicted to lead to S:
      i. $\bar{R} =$ pred. reward for $\bar{S}, \bar{A}, S$
      ii. $P = \bar{R} + \gamma \text{max}_a Q(S', a) - Q(\bar{S}, \bar{A})$
      iii. If $P > \Theta$, insert $(\bar{S}, \bar{A})$ into $PQueue$ with priority P

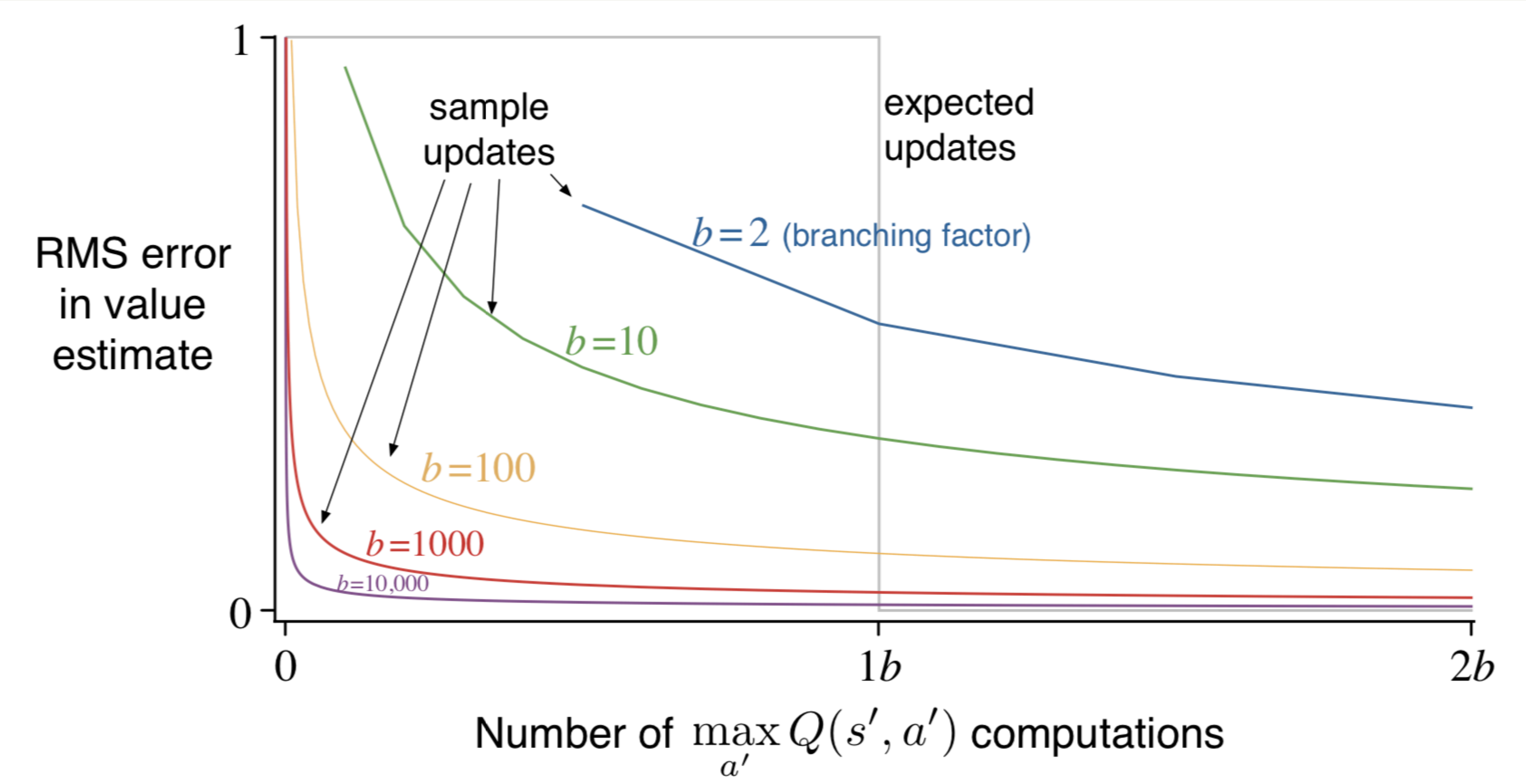
Dimensions

- Update state/action values
- Optimal vs. arbitrary policy
- Expected vs. sample updates



Expected updates > sample updates?

Expected updates > sample updates?



Trajectory Sampling

Sampling:

- Uniform
- According to on-policy distribution

Trajectory Sampling

Sampling:

- Uniform
- According to on-policy distribution

When to plan

- In the background
- At decision time