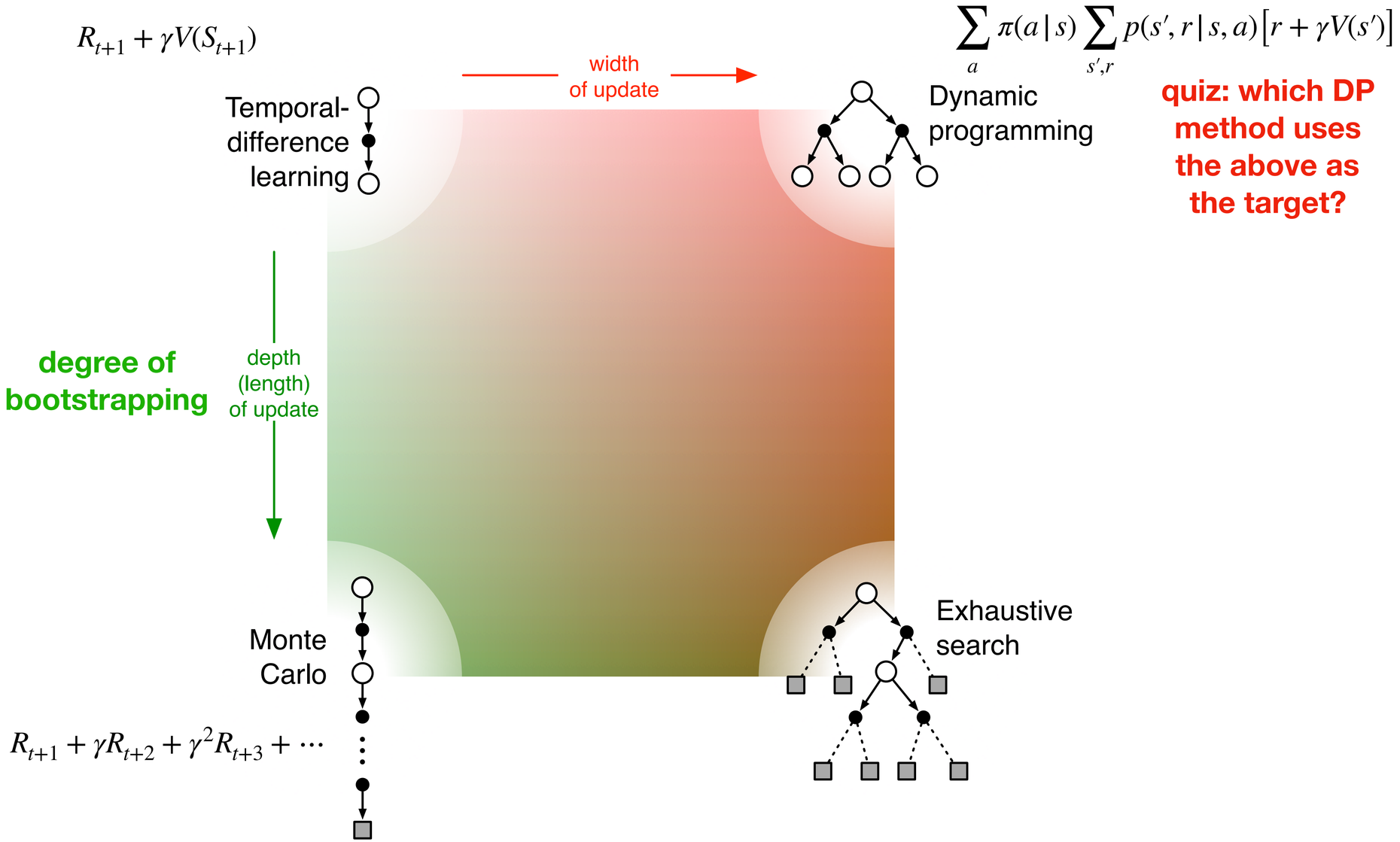
Planning, Learning and Acting

Rupam Mahmood

March 9, 2020

First: Summary of Part 1



Learning and planning

Model of the environment: used by agent to predict environment's response to actions

For example, if the agent has the transition probabilities p(s',r|s,a) as in dynamic programming

Planning: a process that takes model as an input and produces/improves a policy

Dynamic programming methods are planning methods

A planning method uses the model to simulate the environment to produce simulated experience which is used to produce/improve a policy

Model learning: the process of using real experience to improve the model

For example, if we are estimating the transition probabilities based on real experience

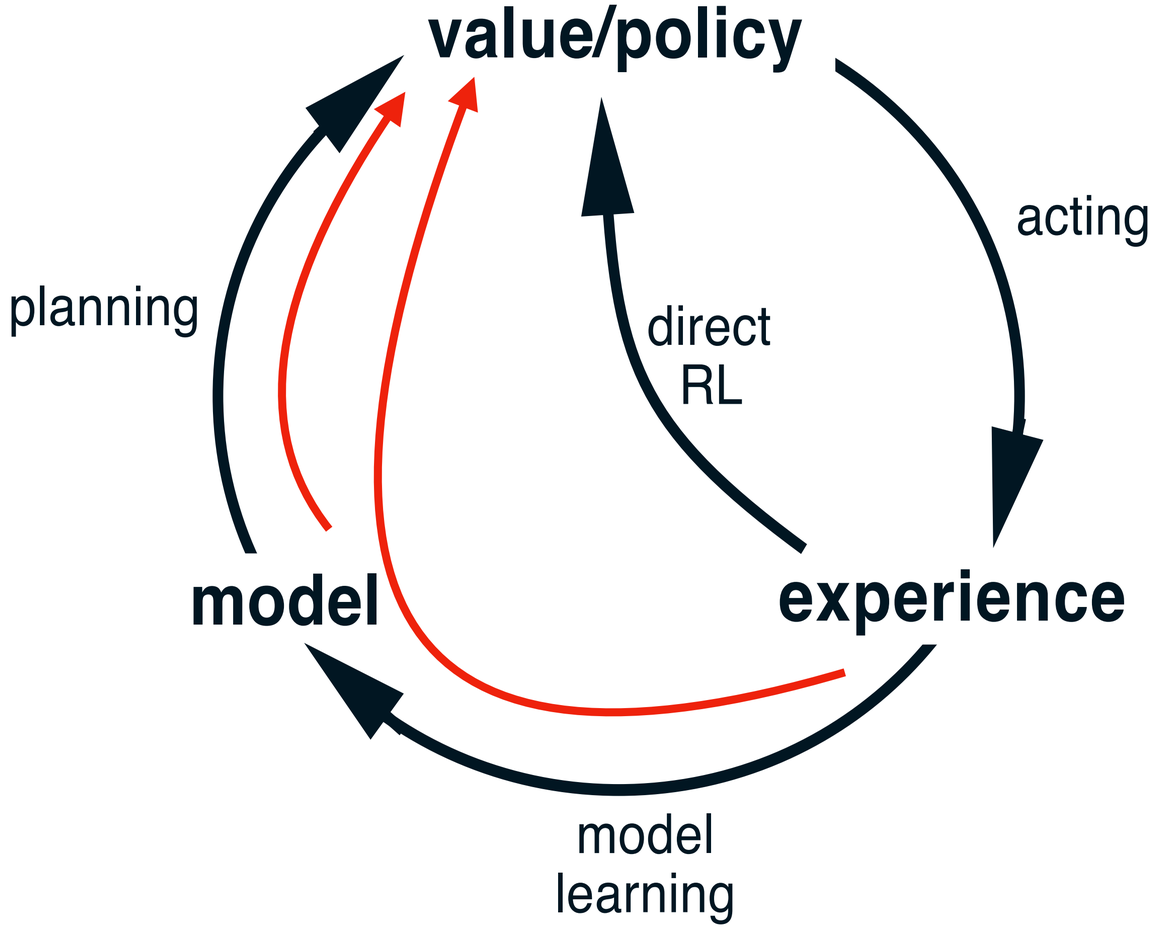
Direct reinforcement learning: the process of using real experience to directly improve the value function and policy

For example, TD or MC methods for prediction and control

Indirect reinforcement learning: the process of using real experience to improve the value function and policy using the model through planning

An architecture for planning, learning and acting at the same time

quiz: indicate indirect RL with an arrow (which of the two red ones?)



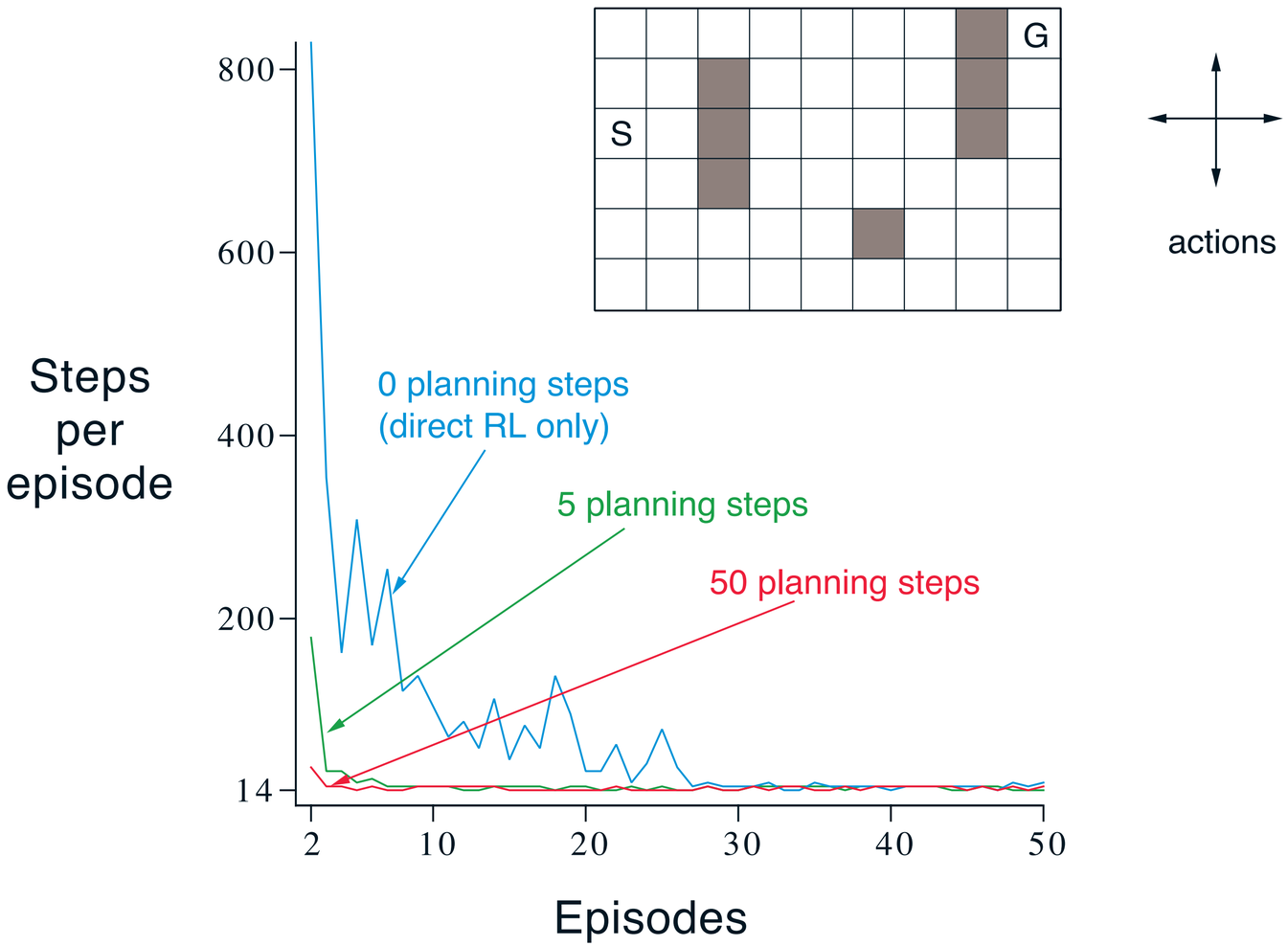
Tabular Dyna-Q

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
  - $R, S' \leftarrow Model(S,A)$
  - $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

Which part is direct RL? What is this method called?

Which part is model learning?

Which part is planning?


G
S
actions
Steps per episode
800
600
400
200
14
0 planning steps
(direct RL only)
5 planning steps
50 planning steps
2
10
20
30
40
50
Episodes

How many planning steps should we take?

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

$R, S' \leftarrow Model(S,A)$

$Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

Dyna-Q+

1. Adds a bonus $\kappa\sqrt{\tau(s,a)}$ to reward in planning

$\tau(s,a)$ denotes the number of time steps (s,a) has not been tried

2. Actions that have not been tried from a previously visited state are allowed to be considered in planning

Where would you put these steps in Dyna-Q to get Dyna-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
  - $R, S' \leftarrow Model(S,A)$
  - $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$

Dyna-Q+: calculating visitation counts

Consider an MDP with one actions (L) and two states with the following episode

S_0	A_0	S_1	A_1	S_2	A_2	S_3	A_3	S_4	A_4
y	**L**	**x**	**L**	**x**	**L**	**y**	**L**	**x**	**L**

Calculate $\tau(s, a)$ for all state-action pairs at each step