Fraunhofer IIS

Fraunhofer Institute for Integrated Circuits IIS

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

[RL22] Q&A Session on MDPs & Dynamic Programming

05.05.2022

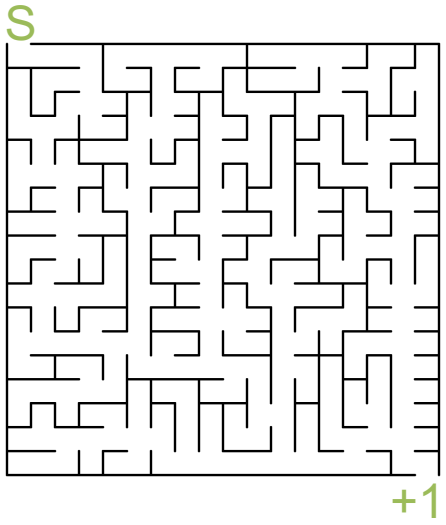
Christopher Mutschler

MDPs

Exercise 3.2 Is the MDP framework adequate to usefully represent *all* goal-directed learning tasks? Can you think of any clear exceptions? □

MDPs

Exercise 3.7 Imagine that you are designing a robot to run a maze. You decide to give it a reward of +1 for escaping from the maze and a reward of zero at all other times. The task seems to break down naturally into episodes—the successive runs through the maze—so you decide to treat it as an episodic task, where the goal is to maximize expected total reward (3.7). After running the learning agent for a while, you find that it is showing no improvement in escaping from the maze. What is going wrong? Have you effectively communicated to the agent what you want it to achieve? □



IID

- **What does ”i.i.d.” mean? (textually)**
  - Independent and indentically distributed

- Why is iid not guaranteed in RL?

  - Independently distributed: Actions have consequences. With the selected actions and the current state, we are in we influence the sample that we see next. Samples within a trajectory are correlated. (aka choices are made according to the trajectory)

  - Identically distributed:
    1. We only see a (very small) subset of the data which is only a rough approximation of the *real* data space
    2. Our behavior policy changes, and we always maximize the target:

$$\mathbb{E}_{\tau \sim \pi}\left[\sum_t \gamma^t R_t\right]$$

this is not stationary!

Policy Iteration

Exercise 4.4 The policy iteration algorithm on page 80 has a subtle bug in that it may never terminate if the policy continually switches between two or more policies that are equally good. This is ok for pedagogy, but not for actual use. Modify the pseudocode so that convergence is guaranteed. □

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
      $\Delta \leftarrow 0$
      Loop for each $s \in \mathcal{S}$:
         $v \leftarrow V(s)$
         $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s))\big[r + \gamma V(s')\big]$
         $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

What happens when two or more actions have the same value?
→ ties broken
→ Keep list of actions and compare