Deep Reinforcement Learning by Aske Plaat

John L. Weatherwax*

November 9, 2024

*wax@alum.mit.edu

To my family.

Introduction

Here you'll find solutions to the problems that I wrote up as I worked through this excellent book. For some of the problems I used `python` to perform any needed calculations or plots. Any code snippets for various exercises can be found at the following location:

`http://www.waxworksmath.com/Authors/N_Z/Plaat/LTP/plaat.html`

I've worked hard to make these notes as good as I can, but I have no illusions that they are perfect. If you feel that that there is a better way to accomplish or explain an exercise or derivation presented in these notes; or that one or more of the explanations is unclear, incomplete, or misleading, please tell me. If you find an error of any kind – technical, grammatical, typographical, whatever – please tell me that, too. I'll gladly add to the acknowledgments in later printings the name of the first person to bring each problem to my attention.

Chapter 3 (Reinforcement Learning):

Questions

Question 1

Most definitions of intelligence include recognition, reasoning, memory, learning, problem solving, and adapting to ones environment.

Question 2

Computer game play focus on learning to compete against another player in a diverse environment.

Question 3

In symbolic AI *reasoning* (i.e. logic) is used to draw facts from a set of underlying facts and assumptions about the world.

In the connectionist AI the inspiration is from biology and the goal is to take smaller units and assemble them into larger units that can do useful things. Success with this method include neural networks.

Question 4

Alan Turing wrote the first chess program (called Turbochamp) but was unable to get the computers of the age to execute it and had to run the algorithm "by hand".

Question 5

In 1997 in New York.

Question 6

Strategy is the high level plan of where to go (or what to do) while tactics are the specific steps to get there.

Question 7

The reinforcement model is one where an agent (the object that learns) interacts with an environment by performing actions in the environment. After each action is performed the agent receives a reward. The goal of the agent is to maximize its total accumulated rewards while interacting in the environment.

Question 8

The 5-tuple of a Markov decision process is (S, A, P, R, γ) where

- S is the set of states that the agent can find itself in
- A is the set of actions that the agent can take in the given states
- P is the probability model of how the state changes after our agent takes each action
- R is a description of the rewards received after each action
- γ is the discount factor indicating how much future rewards should influence the decisions the agent makes *now*. For example if $\gamma = 0$ then no credit for the agents future rewards are given to the agent while if $\gamma \approx 1$ then credit for rewards in the future are given for actions taken now.

Question 9

The value function $V^{\pi}(s)$ represents the "value" (the expected value of all discounted future rewards) of following the policy π when starting in the state s.

Question 10

A policy function $\pi(s)$ is a mapping from states to actions that an agent could take in that state. It can be deterministic or stochastic.

Question 11

Bellman's equation is given by

$$V^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^{\pi}(s') \,,$$

and means that the value starting in state s and following π is the immediate reward we get $R(s, \pi(s))$ plus the discounted expected value of all possible next states s'.

Question 12

After an initial bit of learning we have estimates as to the value of being in certain stats. But *not* estimates of being in other states. The value of being in these other states could be better or worse than the values we have for the known states but we don't know without exploring these states. Whether to allocate resources to explore or to continue to exploit the best known strategy is at the heart of this dilemma.

An ϵ-greedy approach is one such algorithm that trades of exploration and exploitation.

Question 13

Temporal difference learning involves updating our value functions by using the "temporal-difference error" as

$$V(s) \leftarrow V(s) + \alpha(R' + \gamma V(s') - V(s)) \,.$$

Here the temporal-difference error is given by the expression

$$R' + \gamma V(s') - V(s) \,,$$

and represents the error in the value of $V(s)$ once the next action is taken.

Question 14

Approximating a solution is helpful when we need to have the ability to extrapolate our solutions to data/states not seen during the existing episodes.

Question 15

On-policy learning is learning the value function for the same policy that is used to select the actions while off-policy learning is learning the value function for a policy that is *different* than the policy that is generating the actions. Value iteration is an example of an on-policy algorithm and q-learning is an example of an off-policy algorithm.

Question 16

In reinforcement learning the input "data" is an episode and can be computationally demanding to generate. Thus we prefer algorithms that are sample efficient as they will be more computationally efficient.

Algorithms

Exercise 1

These are implemented in the `python` codes `taxi_SARSA.py` and `taxi_qLearn.py`.

Exercise 2

WWX: DP

Chapter 4 (Heuristic Planning):

Questions

Question 1

WWX: Working here.