Homework 01

Yesid Cano Castro, Moritz Lönker and Tim Niklas Witte

Task 01

You are tasked with creating an AI for the game of chess. To solve the problem using Reinforcement Learning, you have to frame the game of chess as a Markov Decision Process (MDP). Describe both the game of chess formally as a MDP, also formalize the respective policy.

Solution

- Set of states $S^{\text{Num rows} \times \text{Num columns} \times \text{Chess pieces}}$.
- Set of actions A: Let be $d = \{\text{up}, \text{down}, \text{right}, \text{left} \dots\}$. $A = \bigcup_{i=1}^{\text{Chess pieces}} \hat{A}_i$ with $\hat{A}_i = \{x : x \in d \wedge \text{isAvaibleAction(x, i)}\}$.
- State dynamics/state transition function $p(s'|s,a) = \text{makeMove(s,a)}$. makeMove(s,a) returns a next state given action a and current state s.
- Reward dynamics $p(R_{t+1}|s,a) = \text{killEnemyPiece(s,a)}$.
  $\text{killEnemyPiece(s,a)} = \begin{cases} 1 & \text{if action } a \text{ does capture a enemy piece in current state } s \\ 0 & \text{otherwise} \end{cases}$
- Initial state $\mu = \text{start state} \in S$.
- Policy: $\pi(s) = \arg\max_{a \in A} V_\pi(s)$ with $V_\pi(s) = \text{Number of captured chess pieces from the enemy}$.

Task 02

Check out the LunarLander environment on OpenAI Gym: Check out this Link!. Describe the environment as a MDP, include a description how the policy is formalized.

- Sate space: set of 8-dimensional sate vectors where each dimension corresponds to:

  - s[0] is the horizontal coordinate
  - s[1] is the vertical coordinate
  - s[2] is the horizontal speed
  - s[3] is the vertical speed
  - s[4] is the angle
  - s[5] is the angular speed
  - s[6] 1 if first leg has contact, else 0
  - s[7] 1 if second leg has contact, else 0

- Action space:
  - Do nothing
  - Fire left orientation engine
  - Fire main engine
  - Fire right orientation engine

- Rewards
  - Moving from the top of the screen to the landing pad and coming. to rest is about 100-140 points.
  - If the lander moves away from the landing pad, it loses reward.
  - If the lander crashes, it receives an additional -100 points.
  - If the lander comes to rest, it receives an additional +100 points.
  - Each lander's leg with ground contact is +10 points.
  - Firing the main engine is -0.3 points each frame.
  - Firing the side engine is -0.03 points each frame.
  - Solved is 200 points.

- Formalize a policy: Since there is a total of four actions, one needs to assign a probability to each of them, for example, 0.25 to each of them. In this case, the system is initialized with a random policy.

- Transition: $p(s'|s, a) = 1$; the probability that action a in state s will lead to state s' is one.

Answers were taken from this repository

Task 03

Discuss the Policy Evaluation and Policy Iteration algorithms from the lecture. They explicitly make use of the environment dynamics ($p(s', r|s, a)$).

- Explain what the environment dynamics (i.e. reward function and state transition function) are and give at least two examples.
- Discuss: Are the environment dynamics generally known and can practi- cally be used to solve a problem with RL?

Solution

The reward function is defined as the expected value of the reward which is gained given a state s, in which the agent performs action a.
Example 1: $r(s, a) = killEnemyPiece(s, a)$ for chess
Example 2: see Task 2 for another Example

The state transition function is defined as the probability of reaching state s' given a state s, in which the agent performs action a.
Example 1: $p(s'|s, a) = makeMove(s, a)$ for chess
Example 2: see Task 2 for another Example

Generally the environment dynamics are not completely known, if they are completely known this is certainly beneficial, as then it would be possible to calculate an optimal policy given enough computing power. In practise this is of course often not applicable, because of the exponential rise in necessary computing power that comes with many applications (e.g. chess).