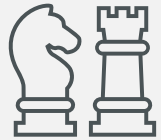
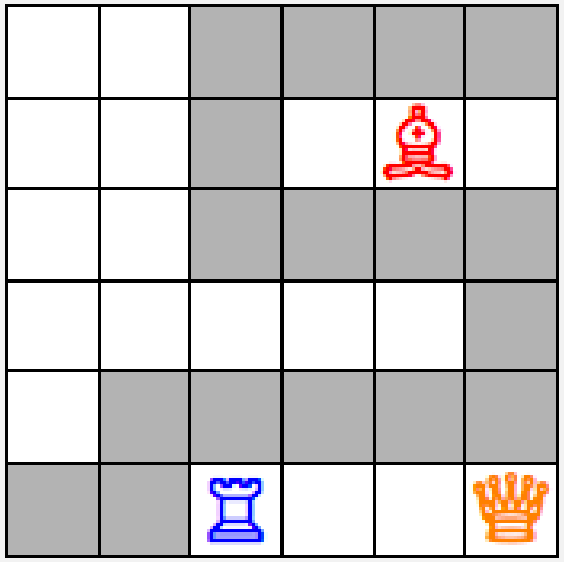


CHESS SNAKE PUZZLES

Reinforcement Learning

Group:
Carolina Cintra – up201906845
Marta Mariz – up201907020
Sara Sá – up201804838

THE GAME AS A MACHINE LEARNING PROBLEM



- Chess Snake Puzzles is a puzzle that mixes the game Snake and the rules of attack of Chess pieces.

- The rules are the following:
  1. The snake can only move orthogonally.
  2. The snake cannot cross a chess piece.
  3. The snake cannot touch itself (even diagonally).
  4. In the result, each piece must attack an equal number of segments of the snake.

- Our work involves trying to win this game by applying reinforcement learning algorithms to contribute for the agent's knowledge of its environment and the rewards it can obtain by taking a specific action in a specific state.

- We will start with a simplified version of the game (4x4 board, only trying to get to the final position) and progress to its complete and final version – including all rules and the appropriate size for the board.

FORMULATION OF THE PROBLEM AS A REINFORCEMENT LEARNING PROBLEM

- State Representation:
  - Pieces have a bitmap with the positions where it can attack the snake
  - The Snake itself has a bitmap of the positions it already passed
  - The board for each level of difficulty is stored in a dictionary in Python
- Algorithms:
  - Q-Learning
  - SARSA
- Rewards:
  - +10 when reaching the final position while all the pieces attack the same number of positions
  - -20 when reaching the end of the game (no possible moves) in a position that is not final
- Operators:
  - Up
  - Down
  - Left
  - Right

REFERENCES

- https://www.youtube.com/watch?v=qhRNvCVVJaA
- https://www.youtube.com/watch?v=mo96Nqlo1L8
- https://deeplizard.com/l
- https://www.geeksforgeeks.org/sarsa-reinforcement-learning/earn/video/mo96Nqlo1L8
- https://www.youtube.com/watch?v=kaDEw5qMTLs&ab_channel=TheLastCode-Bender
- https://www.gymlibrary.ml/#

DESCRIPTION OF THE WORK AND TOOLS

- Jupyter Notebook: for iteractive computing
- OpenAI Gym: for a user-friendly representation of the interaction of the actor and the environment during the development of the algorithms
- PyGame: for a user-friendly representation of the puzzle
- Q-Learning: to learn the value of an action in a particular state
- SARSA: to learn a Markov decision process policy

WORK IMPLEMENTED

- As we have already worked with this puzzle, we have the game logic implemented in Python.

```
def AttackNum(self, snake):
def getPos(self):
def getAttack(self):
def diagonalAttack(self, positions):
def orthogonalAttack(self, positions):
def evalMove(self, move, snake):
def checkPiecesNearby(self):
def countAttacks(self, snake):
```

- The user interface is also already implemented with Pygame.