Homework 3 - Games

Introduction

In this exercise you take the role of a head of one of the two rivaling agencies that try to stop a virus from spreading. You have two main options for stopping the virus: lockdown and vaccination. Your goal is to fare better at stopping the virus than your rival.

Task

As in HW1, the environment you act upon is a rectangular grid, with the same codes for population status. In this exercise, the starting grid will be constant. The grid will also be divided randomly into two equal zones of control for two players in the beginning of the game.

The game consists of two rounds. In the beginning of each round the grid resets to its initial state, and in the beginning of the second round the control zones are switched between the players to ensure symmetry.

Each player has 1 medical and 2 police teams. The round is over when there are no more ‘S’ tiles on the whole map. The performance of each player is measured as described in the ‘Points’ section.

Your task is to create an agent that plays this game and receives as many points as possible.

Actions

Actions remain the same as in HW1, with identical conditions and identical effects, but can only be applied to tiles in the player's own ZOC (zone of control).

Dynamics of the system

The dynamics of the system largely remain unchanged from HW1, with a slight change: first player 1 makes the decision about its action and applies it, then player 2 gets the state **after** the action of player 1 has been applied, and makes the decision about its action, and applies it. Finally, the natural disease spread, end of quarantine etc. happens identically to HW1.

Points

At the end of each turn (after all the effects of actions and natural changes have been applied), score for each player is updated in the following manner:

- Each ‘H’ and ‘I’ tile in player’s ZOC gives +1 point
- Each ‘S’’ tile in player’s ZOC gives -1 point
- Each ‘Q’ tile in player’s ZOC gives -5 points

Code

Your agent resides in hw3.py and you have to implement two functions:

- A constructor `__init__(self, initial_state, zone_of_control, order)` which takes the initial state, the zone of control, and indication whether you go first or second - should finish within 60 seconds
- Function `act(self, state)` - takes in the current state of the game and returns an action. Should finish within 5 seconds.

The code consists of 4 files:

- ex3.py - the file with your agent
- game.py - the file responsible for playing the game
- sample_agent.py - our very simple agent for you to try the game out. You will be given more agents during the following weeks to try for yourself. You also can try to add your own agents as adversaries
- utils.py - the file that contains some utility functions. You may use the contents of this file as you see fit

Submission and grading

You are to submit **only** the file named ex3.py as a python file (no zip, rar etc.). We will run the game against our own agents, and your ex3.py. The check is fully automated, so it is important to be careful with the syntax. The grades will be assigned as follows:

- 50% - if your code wins (on average) against an agent that performs random actions using every team.
- 30% - if your code wins (on average) against our easy agent - an agent that always uses every team with some very simple heuristic. Exact code will not be published, but we will publish some agents more powerful than this one.
- 20% - if your code beats (on average) our intermediate agent - an agent that uses minimax with depth 2 with simple heuristic on the states. Code will not be published
- 20% bonus - if your agent beats (on average) our hard agent.
- The submission is due on the 26.1 (last day of the semester), at 23:59 - no extensions will be given as we are explicitly forbidden to extend the deadline past the semester except for miluim, parents, and hospitalizations.
- Submission in pairs/singles only. You may switch partners from HW1 and HW2 if you wish to do so.

- Write your ID numbers in the appropriate field (‘ids’ in ex3.py) as strings. If you submit alone, leave only one string.
- The name of the submitted file should be “ex3.py”. Do not change it.

Important notes

- We encourage you to double-check the syntax of the output as the check is automated.
- You may use any package that appears in the standard library and the Anaconda package list, however the exercise is built in a way that most packages will be useless.
- You may import utils.py and use any function/class from there
- We encourage you not to optimize your code prematurely. Simpler solutions tend to work best.