CMPE540 – 2022 Spring Homework 2 – Competing killer vacuum cleaners
Due: 17th of May, 23:59

You will implement multi-agent search in this homework. Submit a single .py file.

The program will be called with python3.9 in Linux (Ubuntu) with:

python3 <lastname-firstname.py> <search-type > <init-file> <n-actions>

where

- <search-type> might be one of the following:
  - min-max (no pruning)
  - alpha-beta (pruning in MAX and MIN nodes)
- <n-actions> determines the depth of the search tree
  - Eg. Suppose there are 3 opponent vacuum cleaners which all act optimal:
    - If <n-actions> is 5, the search will be as follows: MAX acts, MIN1, MIN2, MIN3 acts, MAX acts, and the search stops, and the utility values after the second MAX action will be used.
- <init-file> will be a text file gives all details related to the initial environment.

Submission Instructions:

- Please follow these instructions, otherwise you will lose significant points.
- The project will be automatically evaluated. Still, you need to prepare a very short report. See the end of this document for more information about what is expected in the report.
- Name your file with your lastname (lastname.pdf lastname.py)

Late Submission:

- - (10 x days) upto 3 days
- In case there is serious mistake in the project description or significant change, this date will be postponed.

Discussion and Cheating:

- I don't like this section, but still should write about this..
- You are welcome to discuss the topics about the project. But please do not cheat. There are programs that can automatically check the similarity of the source code and/or execution. Read the department policy about what is considered as cheating. If I am convinced that you cheated, then you will get F and disciplinary action is filed.

Project objectives:

- In this project, you will implement multi-agent search

Introduction to the problem:

The environment is as follows:

- The environment is NxM grid world.
- Each grid in the environment might contain:
  - Vacuum cleaner (our agent)
  - Enemy vacuum cleaners
  - Obstacles that avoid entering to that grid. There is not dirt in the obstacle with grid.
  - One dirt
- The vacuum cleaners have 6 actions:
  - left, right, up, down moves the cleaner one grid, unless that grid is an obstacle.
  - suck action that sucks one dirt.
  - stop action does nothing.

- The environment, agent type, locations of the obstacles, dirt, vacuum cleaners (our agent and other agents) will be provided in a text file.
- Tie-breaker:
  - If required, the precedence used as a tie-breaker is as follows: left, right, down, up, stop, suck
- Opponent vacuum cleaners, which are numbered with even digits, move randomly
- Opponent vacuum cleaners, which are numbered with odd digits, move optimally
- Your vacuum cleaner starts first, then other vacuum cleaners (ordered by their digits) move next to each other.

Utility function:

The utility value at any node is calculated as follows:

- If your vacuum cleaner is in the same grid with one of the any other opponent, utility is set to -100 and the episode ends.
- Otherwise, utility = (the number of dirts cleaned by your cleaner – the number of dirts cleaned by your opponents)

Example input file:

```
xxxxxxxxxx
x  1   2 x
x . x  . x
x   x  4 x
x c x  . x
xxxxxxxxxx
```

- where
  - x corresponds to obstacles
  - c corresponds to your vacuum cleaner
  - each <digit> corresponds to one of your opponents where
    - even <digit> opponents move randomly
    - odd <digit> opponents move optimally
  - . (dot) corresponds to the dirt

Output:

- After running the search, you need to print out the following (to standard output):

Action: <action>
Value: <value>
Util calls: <n-util-calls>

where
• <action> is the optimal action of your agent in its first move
• <value> is the (expected) minimax value of the root node
• <n-util-calls> is the number of calls for utility function