Programming Homework 11

CS540

August 9, 2019

1 Instruction

Please submit your output files and code on Canvas → Assignments → P11. Please do not put code into zip files and do not submit data files. The homework can be submitted within 1 weeks after the due date on Canvas without penalty (50 percent penalty after that).
Please add a file named "comments.txt", and in the file, you must include the instructions on how to generate the output, for example:

- Data files required: train.csv, test.csv. Run: main.jar.
- Data folder required: data/train1.png ... data/train100.png . Compile and Run: main.java.

2 Details

All the requirements are listed on the course website. The following is only an example workflow to solve the problem.

1. Create a method to check if the game is over and assign the winner. Player X wins if there are three X in a row or column or diagonal. Player O wins if there are three O in a row or column or diagonal. If no one wins and no cell on the board is empty, then the game ends in a tie. You can store the board configuration in a string of X and O or a integer array of $1, 0, -1$.
2. Create State class storing the following variables.
   - The board configuration.
   - The backtracker keeping track of the order in which the symbols are placed.
   - The name of the player, either X or O.
   - The α or β value at this state, either -1 or 0 or 1.

   For example, if X is placed at position 0, then O is placed at position 1, then X is placed at position 2, then O is placed at position 4, then the board configuration should be $XOX?O????$ (where ? represents a blank cell), the backtracker is 0124, and the player is X. The $\alpha\beta$ values will be computed in the next step.

3. Create a recursive method to compute the α and β values. The recursion should look at like the following, given the current State s and the current player is $i \in \{X, O\}$. Function name: $v(s) \rightarrow \{-1, 0, 1\}$

   Base case:

$$\text{return} \begin{cases} 1 & \text{if } X \text{ wins at } s \\ 0 & \text{if tie at } s \\ -1 & \text{if } O \text{ wins at } s \end{cases}$$

   Recursive case:

$$\text{return} \begin{cases} \max\limits_{s' \in s'(s)} v\left(s'\right) & \text{if } i = X \\ \min\limits_{s' \in s'(s)} v\left(s'\right) & \text{if } i = O \end{cases}$$

4. Create another recursive method to output all optimal paths. Suppose the value of the whole game (the value at the root state in the previous step) is $v^\star$. Function name: $out(s)$

   Base case:

$$\text{if } s \text{ is a terminal state , print backtracker}$$

   Recursive case:

$$\text{for each } s' \in s'(s) \text{ return} \begin{cases} out(s') & \text{if } v(s') = v^\star \\ \text{do nothing} & \text{if } v(s') \neq v^\star \end{cases}$$