Project description

In a typical zero sum two player game, players are generally competing for a certain common resource, and their gain is a function of their share of the resource. Often players have other challenges such as satisfying other constraints on other personal resources such as time, energy or computational power in the course of the game. In this project, I introduce ***The Fruit Rage!*** a game that captures the nature of a zero sum two player game with strict limitation on allocated time for reasoning.

The task was creating a software agent that can play this game against a human or another agent.

Rules of the game

The Fruit Rage is a two player game in which each player tries to maximize his/her share from a batch of fruits randomly placed in a box. The box is divided into cells and each cell is either empty or filled with one fruit of a specific type.

At the beginning of each game, all cells are filled with fruits. Players play in turn and can pick a cell of the box in their own turn and claim all fruit of the same type, in all cells that are connected to the selected cell through horizontal and vertical paths. For each selection or move the agent is rewarded a numeric value, which is the square of the number of fruits claimed in that move. Once an agent picks the fruits from the cells, their empty place will be filled with other fruits on top of them (which fall down due to gravity), if any. In this game, no fruit is added during game play. Hence, players play until all fruits have been claimed.

Another big constraint of this game is that every agent has a limited amount of time to spend for thinking during the whole game. Spending more than the original allocated time will be penalized harshly. Each player is allocated a fixed total amount of time. When it is your turn to play, you will also be told how much remaining time you have. The time you take on each move will be subtracted from your total remaining time. If your remaining time reaches zero, your agent will automatically lose the game. Hence the player should think about strategies for best use of his/her time (spend a lot of time on early moves, or on later moves?)

The overall score of each player is the sum of rewards gained for every turn. The game will terminate when there is no fruit left in the box or when a player has run out of time.

Game setup and examples

Figure 1 depicts a sample 10 x 10 game board with 4 types of fruits denoted by digits 0, 1, 2 and 3 in the cells. By analyzing the game, the agent should decide which location to pick next. Let’s assume that it has decided to pick the cell highlighted in red and yellow in figure 1.

Figure 2 shows the result of executing this action: all the horizontally and vertically connected fruits of the same type (here, the selected fruit is of type 0) have been replaced by a * symbol (which represents an empty cell). The player will claim 14 fruits of type 0 because of this move and thus will be rewarded 14^2 = 196 points.

```
---------------------
|3|1|0|2|3|2|2|3|1|0|
|0|1|2|1|2|3|2|0|1|3|
|3|0|2|1|1|1|1|1|1|3|
|0|2|2|1|0|3|1|1|3|2|
|0|2|3|0|0|1|1|0|1|2|
|0|3|2|3|3|2|1|0|1|0|
|2|0|0|3|0|2|2|0|1|2|
|2|2|0|2|2|0|0|0|2|1|
|0|1|3|0|0|0|0|0|2|0|
|2|2|0|0|0|2|2|2|3|1|
---------------------
```

figure 1.

```
---------------------
|3|1|0|2|3|2|2|3|1|0|
|0|1|2|1|2|3|2|0|1|3|
|3|0|2|1|1|1|1|1|1|3|
|0|2|2|1|0|3|1|1|3|2|
|0|2|3|0|0|1|1|*|1|2|
|0|3|2|3|3|2|1|*|1|0|
|2|0|0|3|0|2|2|*|1|2|
|2|2|0|2|2|*|*|*|2|1|
|0|1|3|*|*|*|*|*|2|0|
|2|2|*|*|*|2|2|2|3|1|
---------------------
```

figure 2.

```
---------------------
|3|1|*|*|*|*|*|*|1|0|
|0|1|0|*|*|*|*|*|1|3|
|3|0|2|2|3|2|2|*|1|3|
|0|2|2|1|2|3|2|*|3|2|
|0|2|2|1|1|1|1|*|1|2|
|0|3|3|1|0|3|1|3|1|0|
|2|0|2|0|0|1|1|0|1|2|
|2|2|0|3|3|2|1|1|2|1|
|0|1|0|3|0|2|2|1|2|0|
|2|2|3|2|2|2|2|2|3|1|
---------------------
```

figure 3.

```
---------------------
|3|1|*|*|*|*|*|*|1|0|
|0|1|0|*|*|*|*|*|1|3|
|3|0|2|2|3|2|2|*|1|3|
|0|2|2|*|2|3|2|*|3|2|
|0|2|2|*|*|*|*|*|1|2|
|0|3|3|*|0|3|*|3|1|0|
|2|0|2|0|0|*|*|0|1|2|
|2|2|0|3|3|2|*|*|2|1|
|0|1|0|3|0|2|2|*|2|0|
|2|2|3|2|2|2|2|2|3|1|
---------------------
```

figure 4.

```
---------------------
|3|1|*|*|*|*|*|*|1|0|
|0|1|0|*|*|*|*|*|1|3|
|3|0|2|*|*|*|*|*|1|3|
|0|2|2|*|3|*|*|*|3|2|
|0|2|2|*|2|2|*|*|1|2|
|0|3|3|2|0|3|*|*|1|0|
|2|0|2|0|0|3|2|*|1|2|
|2|2|0|3|3|2|2|3|2|1|
|0|1|0|3|0|2|2|0|2|0|
|2|2|3|2|2|2|2|2|3|1|
---------------------
```

figure 5.

Figure 3 shows the state of the game after the empty space is filled with fruits falling from cells above. That is, for each cell with a * in figure 2, if fruits are present above, they will fall down. When a fruit that was on the top row falls down, its previous location is marked as empty (i.e., it becomes a * symbol). That is, no new fruits are injected to the top of the board. In addition to returning the column and row of your selected fruit, the agent will also need to return this resulting state after gravity has been applied. The game is over when all cells are empty, and the winner is determined by the total number of points, that is, sum of [fruits taken on each move]^2 (it is possible to end in a draw if both players score the same).

In figure 3, the opponent player then decided to pick the location highlighted in green and yellow. Upon selecting this cell, all 12 fruits of type 1 connected to that cell will be given to the opponent player and thus the opponent player will gain 12^2 =

144 points. In figure 4, cells connected to the selected cell are marked with * and in figure 5 you see how some of those picked fruits are replaced with the contents of cells above (fruits above fell down due to gravity).

For this AI agent, I implemented a **minimax algorithm with alpha-beta pruning**.

Input: The file `input.txt was` formatted as follows:

First line: integer n, the width and height of the square board (0 < n <= 26)

Second line: integer p, the number of fruit types (0 < p <= 9)

Third line: strictly positive floating point number, (your remaining time in seconds)

Next n lines: the n x n board, with one board row per input file line, and n characters (plus end- of-line marker) on each line. Each character can be either a digit from 0 to p-1, or a * to denote an empty cell.

Output: The file `output.txt` was formatted as follows:

First line: the selected move, represented as two characters: **A letter** from A to Z representing the column number (where A is the leftmost column, B is the next one to the right, etc.), and **A number** from 1 to 26 representing the row number (where 1 is the top row, 2 is the row below it, etc.).

Next n lines: the n x n board just after your move and after gravity has been applied to make any fruits fall into holes created by your move taking away some fruits (like in figure 3).