Game of Choice

Eckel, TJHSST AI1, Fall 2022

Background & Explanation

We played Tic-Tac-Toe, Othello, and an additional game in the Modeling Challenge. Now, it's up to you – extend your knowledge from unit 3 by modeling whatever game you choose, making it playable, and adding AI.

Depending on your choices, this assignment may not progress in the usual GREEN > BLUE > RED > BLACK way; you might want to work on the requirements in a different order, or simultaneously. Read the whole document, think about the time you have available, and make a plan that works for you.

THIS ASSIGNMENT MAY BE DONE WITH A PARTNER! If so, I will expect a more complex game; be sure to run your choice by me first before you begin coding.

GREEN Credit: Choose and Model a Game

You can't choose just any game; it has to be more complicated than Othello. It also probably needs to be adversarial – two or more players pitted against each other making alternating moves. (Unless you have some weird clever idea; I'm always happy to hear it.) It does *not* need to be symmetric; there are many games where the two players are doing different things and that's totally fine too.

That still leaves your choice pretty wide open!

For GREEN credit, all you have to do is model it and make it playable by one human against another. That said, if you're planning on going past GREEN credit, you probably want to read the later sections of the assignment before you make your choice.

Here are some definitely good choices:

- Ultimate Tic-Tac-Toe is a great one; if you've never heard of it go to http://bejofo.net/ttt and play a few games.
- Chess
- Go
- Hive
- A bunch of games in the Math Games with Bad Drawings book – feel free to take a look; I especially like Sequencium, though it might be a challenge to show in console output.
- Anything else you want to propose! Just ask me first.

Here are some definitely bad choices:

- Don't pick Checkers. It's too simple.
- Don't pick a game that's solved, like Nim.
- Don't pick Battleship or anything else where an AI would just consist of some kind of consistent pre-programmed pattern.

Specification for GREEN Credit

Submit a single Python script to the link on the course website.

This assignment is **complete** if:

- You follow the instructions on the submission form to format your submission properly.
- When I run your code, it explains the game and how to play. Then, two players alternate turns. Your code doesn't allow illegal moves, and knows when the game ends.

BLUE Credit: Use minimax and A/B pruning to add AI

Take the game you've modeled and add AI.

Specifically:

- Create three AI players:
  - RANDOM should make a random valid move.
  - AGGRESSIVE should always try to capture the most important enemy piece it can, or achieve the biggest score gain it can, etc. If it can't capture anything, it plays as random.
  - BEST should be your best implementation of Minimax, Alpha/Beta, etc. It should beat the previous two players the vast majority of the time.
- Write Python code that takes two command line arguments from the set of "RANDOM", "AGGRESSIVE", "BEST", "USER". It should then play out a game between the two selected players, with the player identified in the first argument going first. If "USER" is selected, instructions for how to input moves myself should be clear. Your code should not allow illegal moves, as before.
- Write a *BRIEF* report (this really should not take long) where you give a sketch of how your strategy works. Two to three paragraphs is plenty.

Specification for BLUE Credit

Submit your Python script and your report (two files) to the link on the course website.

This assignment is **complete** if:

- You follow the instructions on the submission form to format your submission properly.
- Your Python file meets the above specification for input and plays each game accordingly.
- Your report contains the information I need.
- Runtime is reasonable (ie, I don't have to wait a ridiculously long time for the AI to make a move). Keep each move under approximately 10 seconds if possible; if not, let me know and we can negotiate.

RED Credit: Add Advanced AI

This is pretty wide open. Two requirements:

- You need to invent or find an AI technique that I never taught you to make your AI stronger. There are some ideas on the Othello assignment, some of which are Othello-specific and some that aren't. You can also probably find something if you google your game, especially if it's well known.
- You need to add difficulty levels. That is, you now need to have these options at runtime:
    - USER – the user
    - AGGRESSIVE – a dumb AI as described above; it does not look into the future
    - ONE – an AI that uses minimax, but is not as difficult to beat as…
    - TWO – an AI that uses minimax, but is not as difficult to beat as…
    - THREE – your hardest AI to beat.

  It's up to you how you differentiate between levels one, two, and three!

As with BLUE, I'll want you to write a quick report telling me what you added – explain your new technique and explain how you made different difficulty levels.

Specification for RED Credit

Submit your Python script and your report (two files) to the link on the course website.

This assignment is **complete** if:

- You follow the instructions on the submission form to format your submission properly.
- Your Python file meets the above specification for input and plays each game accordingly.
- Your report contains the information I need.
- Runtime is reasonable (ie, I don't have to wait a ridiculously long time for the AI to make a move). Keep each move under approximately 10 seconds if possible; if not, let me know and we can negotiate.

BLACK Credit: Add GUI

Use a Python graphics library to make your game interactive with a graphical user interface. If you want to submit it to me, you have to use tkinter – the built in Python graphics library. (Google how to use it, or get some ideas from the tkinter demo included with the Train Routes assignment.) If you do something else, you might have to show me in person, and that means you'll have to be done a little further in advance.

Your GUI should include the ability to choose from the options given in the RED credit section without using command line inputs.

Specification for BLACK Credit

Submit your Python script to the link on the course website (or, if you didn't use tkinter, finish early and show me in person.)

This assignment is **complete** if:

- You have a great GUI for your game, and you meet all prior requirements.
- Runtime is reasonable (ie, I don't have to wait a ridiculously long time for the AI to make a move). Keep each move under approximately 10 seconds if possible; if not, let me know and we can negotiate.