CONTENTS

Update Mar 22: added starter code

Cmput 455 Assignment 4

Due Apr 8, **11:55pm** (two days extension compared to the outline). Submit via eClass submission link on the main eClass course page.
Late submission (with 20% deduction) Apr 10, 11:55pm. Submit via separate eClass link for late submissions.

In this assignment, you develop your own player for the game of NoGo. You can implement any player you want using any techniques, as long as it's in Python3 and follows the rules and constraints below. For example you can use minimax or MCTS or both, add an exact endgame solver, add heuristic evaluation, move ordering, simulations, etc. We will test the performance of your player in two tournaments.

- In the **marking tournament**, for regular assignment marks, we will match your player against a fixed set of our NoGo players.
- The **championship tournament**, for bonus points and eternal fame, will be a tournament among student programs. The winning team will be declared the Cmput 455 NoGo champions and win small prizes.

Setup

1. First, as in previous assignments, make sure you have your Python 3, NumPy and GoGui 1.51 set up.
2. Download assignment4.tgz and expand it. The directory `assignment4` contains:
   - Code for two of the four opponents for the marking tournament: a `random_player` directory containing the random NoGo player `nogo_random.py`, and a `flat_mc_player` directory containing a simple simulation-based player `nogo_flat_mc.py`.
   - A script `play.py` that you can use for playing test games between two programs. For an example of how to use `play.py`, see the presubmission instructions below.
   - A directory nogo4 which you will use to implement your player. Your player is called `nogo4.py`. Right now this player is just a copy of flat_mc_player, but in this assignment you replace this simple playing engine with your own, better one.
   - We will not publish the other two players before the tournament, but here is a little bit of information about them. One will be an optimized MCTS-based player, and the other one will be a mystery opponent. We will tune the strength of these two players such that they will be a little bit more challenging than the simulation player, but not impossible to defeat within the scope of an assignment.
3. Modify/add code only in directory `assignment4/nogo4` to implement your solution.

RULES

- All games will be played on a 7x7 board on a standard lab machine.

- The time limit will be 30 seconds per move. If a program does not play within the time limit, it will be killed by the script and instantly loses the game. We recommend leaving a little bit of extra time and not fully use the 30 seconds, just to be safe.
- The memory limit will be 1 Gigabyte per program.
- If a program generates an illegal move, it instantly loses the game.
- If a program crashes or exceeds the memory limit, it instantly loses the game.
- If a game is interrupted for other reasons, it may be replayed at the discretion of the tournament directors Martin Müller and Ting-han Wei.
- Your player must send resign in reply to a `genmove` command if there are no legal moves. The random and flat_mc players already implement this functionality.

CONSTRAINTS

- Teams are strongly encouraged to practice social distancing measures such as working together remotely.
- You can use any code provided by us or created by you as part of this course. You can use code libraries that are part of the standard Python3 distribution as exists on the lab machines. You cannot use other outside sources of code.
- Your playing engine is not allowed to use more than one thread.
- Your program is not allowed to use programming languages other than Python3.
- Your program is allowed to read/write files within your `assignment4/nogo4` directory only.
- The total size of an assignment submission, including all files, is limited to 1 Megabyte uncompressed.
- Further reasonable constraints to prevent abuse may be imposed as we become aware of them.

GTP COMMANDS

- `genmove color`

  Your `genmove` command should generate a move using your player, and comply with the rules and constraints above.
- All other existing GTP commands should be left as-is.

The Usual Warnings, Hints and Details - Read them All

- Please make sure that these details are correct in your assignment 4 submission.
- Your file must be a valid tgz file named `assignment4.tgz` which can be uncompressed with `tar -xf assignment4.tgz`
- Keep the name of your player `nogo4.py` and keep it in the `nogo4` subdirectory, to allow our scripts to find it.
- Do not introduce extra levels of directories or different directory names.
- You may add extra python files within your `assignment4/nogo4` directory only.
- By default, we assume that teams will stay the same as in assignment 3. If you change your team, email Chao with the new information before the assignment 4 deadline.
- **NEW - Team Names**: We would like to provide updates as the tournament progresses. In your readme file, please provide a teamname as well, which we will use for public tournament updates. The team name should provide a level of anonymity.

Pre-submission Test and Submission

Play one test game against `nogo_random.py` as follows:

- In `play.py`, change player1 to your program's path, `nogo4/nogo4.py`, and player 2 to the random player
- Use numGame=1 as the parameter for the `playGames()` function.
- Run `play.py`

- After the game finishes, there will be a file named `game_result.txt`, that contains the summary of the test game(s). For presubmission, do both: Log the testing procedure as in previous assigments, and submit this `game_result.txt` file as well.

Follow the same general steps as in assignment 1 to create your `presubmission.log` file and your submission, but (of course) using your `assignment4` directory, `assignment4.tgz` as file name, and playing the sample test game described above as your presubmission test. Add your `presubmission.log`, `game_result.txt` and `readme.txt` to your `assignment4` directory at the top level (NOT inside nogo4). Your readme file should also state your **public team name** for the tournament.

Marking

There will be 5+2 marks for this assignment.

- 1 mark total for your three text files `presubmission.log` which shows the log of your correct test execution, your `game_result.txt`, and your `readme.txt`.
- 4 marks for your wins against our 4 test opponents. There will be two games per opponent, one with each color. The mapping of wins to marks will be decided by the instructors, since it is hard to predict results against our players.
- Up to 2 bonus marks for doing well in the championship tournament.

Created: Mar 12, 2020 Last modified: Mar 21, 2020

Martin Müller
