The Game of Yinsh (Phase I)

COL333

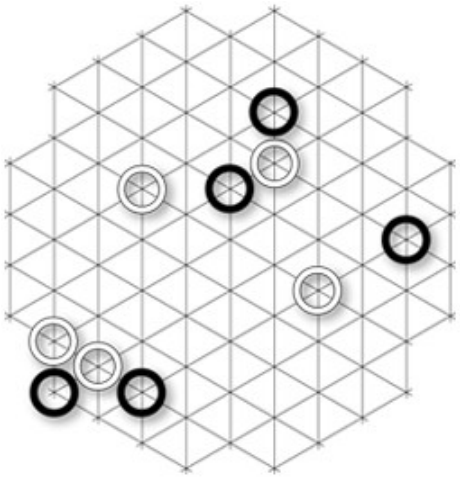
August 30, 2018

1 Goal

The goal of this assignment is to learn the adversarial search algorithms (minimax and alpha beta pruning), which arise in sequential deterministic adversarial situations.

2 The Game of Yinsh (N,M,K,L)

2.1 Setup



The game board is almost a regular hexagon of side N as shown. There are M rings with each player. Typical values for N are 5 and of M is also 5.

2.2 Objective

There are 2 players, Black and White. The objective for both the players is to remove L rings before the other player does. To remove a ring you must create K markers in a line (explained below).

2.3 Player Moves

The game proceeds turn by turn. The first M moves for both the players are placing the M rings each player has on the board on a position of their choice that is not already occupied. The subsequent moves are either moving a ring or removing a ring when you have at least 5 markers in a row. Moving a ring leaves a marker of the ring's color behind.

2.3.1 Moving a ring

You can move a ring from its initial position to another position on the board, but it can't cross any other ring. If you are moving across markers, you can only move to the first empty space after the first continuous line of markers. Any number of spaces can be skipped before the first continuous line of markers. Any number of spaces can also be skipped if there are no markers on the line. In other words, you are allowed at most one (possibly long) jump across the markers. When a ring crosses a marker, the color of the marker toggles (black to white and white to black). This is applicable for all markers the ring crosses.

2.3.2 Removing a ring

If you get 5 markers in a row, you can remove them by clicking on one of the end markers of the 5 in a line. If there are more than 5 in a line or two 5 lines crossing each other, clicking at the two ends will remove the 5 markers in the middle. After removing 5 continuous markers, you can remove a ring of your choice. Find more details in the link to the rules above. Catch : unlike in the rules, where you can remove any 5 continuous markers of yours choice, you can only remove 5 continuous markers from one end.

Rules : `http://www.gipf.com/yinsh/rules/rules.html`

3 Performance Equivalence Classes

To rank your performance we will first divide each game into a set of equivalence classes:
Win > Stalemate win > Draw > Stalemate Loss > Loss

3.0.1 Win / Loss

You remove L (usually 3) rings before the other player does. The other player loses. For winning, 3-0 is better than 3-1 which is better than 3-2. The opposite is for losing.

3.0.2 Stalemate Win / Stalemate Loss

When no more legitimate moves are possible, the player with more rings outside the board wins. The opponent loses the stalemate. For stalemate win, 2-0 is

better than 2-1 and 1-0 (same bracket). The opposite for losing.

3.0.3 Draw

When no legitimate moves are possible and both the players have the same number of rings outside the board. For draw, 2-2 is better than 1-1 which is better than 0-0.

3.0.4 Final Ranking

Within each equivalence class, the performance will be ranked by the difference of the number of markers (you - opponent) on the board at the time the game ended. Each pair of players will play 2 games, each getting a chance to start first in one game. The final ranking will be decided by combining the ranking of all the games.

4 What is being provided?

Your assignment packet contains code for the game server in python and starter code for your player in Python that: (1) interacts with the game server, (2) allows you to manually send moves to the server. At the server end, you are provided a GUI which allows you to visualize the proceedings of the game.

The server code can be found at: `https://github.com/NikhilGupta1997/Yinsh-AI`

5 Interaction with the Game Server

Note that you will require Python 2.x. The server won't work with Python 3.x.

5.1 Dependencies

Python 2.x dependencies : pip install selenium jinja2 numpy
Install chromedriver : `http://chromedriver.chromium.org/downloads`
Unzip and copy it to someplace in your PATH variable. Eg : /usr/bin

5.2 Running the server

The server that you are given will run via the command:
python server.py <port> <max-clients>
Make sure to cd into the server folder. You are also given a config.txt file, where you can specify the same arguments as above, if you would like to fix them (i.e. port and max-clients). (python server.py 10000 -n 5 -NC 2 -TL 120 -LOG server.log)

5.3 Running the client

cd into the client folder
python client.py <port-no> <server-ip>
Open the server and client from two different terminals. The server will allow <max-clients> number of clients to connect to it.

6 What to Submit?

6.1 Submission Files

Submit your code for your game player. The code should be contained in zip file named in the format `<EntryNo>.zip`. If there are two members in your team it should be called `<EntryNo1>_<EntryNo2>.zip`. Make sure that when we unzip the following files are produced:

1. `compile.sh`
2. `run.sh`
3. `writeup.txt`

You will be penalized for any submissions that do not conform to this requirement. Your code must compile and run on our VMs. They run amd64 Linux version ubuntu 12.04. You are already provided information on the compilers on those VMs. These configurations are similar to GCL machines like 'todi' (have a RAM of 16 GB).
The `writeup.txt` should have two lines as follows

1. First line should be just a number between 1 and 3. Number 1 means C++. Number 2 means Java and Number 3 means Python.

2. Second line should mention names of all students you discussed/collaborated with (see guidelines on collaboration vs. cheating on the course home page). If you never discussed the assignment with anyone else say None.

After these first two lines you are welcome to write something about your code, though this is not necessary.

6.2 Code Verification

Your submission will be auto-graded. This means that it is absolutely essential to make sure that your code follows the input/output specifications of the assignment. Failure to follow any instruction will incur significant penalty. The details of code verification will be shared on Piazza (similar to A1).

7 Evaluation Criteria

There are two evaluation phases to this Assignment.

7.1 Phase1 (A2)

1. This phase will be worth 5 marks.
2. The submitted bot will be tested and scored, once as orange and once as blue, against 3 bots made by the TAs. That is a total of 6 games.
3. Each game will be given a score (Look at the scoring section above). The final score will be the sum of scores from the 6 games.
4. A seed for each bot will be created based on this final score for the next phase. In case of a tie between the final scores of two bots, the tiebreak will be determined by the sum of the scores of the TA bots.

7.2 Phase2 (A5)

1. This phase will be worth 15 marks.
2. The submitted bots will be placed in a tournament tree based on their seeds (Inspiration taken from a Wimbledon Tournament Tree). The tree will initially consist of a round robin stage (4-6 bots in each stage) followed by a knock-out tournament.
3. The scores for each bot will be based on how far it progresses in the tournament. This score distribution will be revealed in the future.
4. Each tournament will be worth 5 marks and a total of 3 tournaments will be played. The winner of each tournament will be removed from the subsequent tournaments and will receive full 15 marks.
5. More details on this shall be notified at the time of assignment 5.

8 What is Allowed / Not Allowed?

1. You may work in teams of two or by yourself. If you work in a team of two then make sure you mention the team details in the write-up. Our recommendation – this will lead to a much more open ended final assignment (phase 2); hence best to work with a partner with whom you communicate well. You will be allowed to use the same partner for the final project. Also, you cannot use a partner from the previous assignment.
2. While you are allowed one of three languages we recommend that you use C++ since it produces the most efficient code. This assignment requires you to play the best game within a given time constraint.
3. Your code must be your own. You are not to take guidance from any general purpose AI code or problem specific code meant to solve this or a related problem. This means **no guidance from codebases of other game players whether of this game or another, and no guidance**

from pre-written feature learning codes. However, you will be allowed to use some machine learning libraries if you wish to use them. Kindly make specific (public) requests on Piazza and we will approve on a case by case basis.

4. It is preferable to develop your algorithm using your own efforts. However, we will not stop you from google searching for ideas.

5. You must not discuss this assignment with anyone outside the class. Make sure you mention the names in your write-up in case you discuss with anyone from within the class outside your team. Please read academic integrity guidelines on the course home page and follow them carefully.

6. You get a zero if your player does not match with the interaction guidelines in this document.

7. We will run plagiarism detection software. Any team found guilty will be awarded a suitable penalty as per IIT rules.