ASSIGNMENT 5: ENTROPY (Phase 2)

Objective: We will play three tournaments on Entropy(N), with the three configurations of N: N=5, N=6, and N=7. Each tournament will involve various game engines competing against each other. Each match will comprise some sets of different (unknown) tile sequences. In each set two games will be played changing the roles of order and chaos.

As a reminder here are the rules to compute the margin of victory:

1. If a complete set happens without timeouts and invalid moves, then the score of the winner will be winner's score as order – loser's score as order. The loser's score will be negative of this.

2. If the set didn't complete because of one player timing out or making an invalid rule the following rules will apply:

(a) If player 1 was playing as order and the opponent makes a mistake, player 1 will be assumed to have scored M in this game.

(b) If player 1 was playing as chaos and the opponent makes a mistake, the opponent's score will be zero in this game.

Computation of M

The exact value of M will be decided after all matches in round-robin are finished. M will be calculated as the maximum score obtained by any player in any game in this set.

Tournament scoring

- A team A will win a match against another team B if its average score over all sets is greater than zero. If it is zero, the team winning more sets will be declared the winner. If we can still not decide, the match will be considered drawn. The winning team will get 3 tournament points. If the match is drawn, both teams will get 1 tournament point each.

- The top team in a group will be decided as the team which has the maximum tournament points. If there is a tie, the team who won the match between the two tied teams will go the next round. If it is still a tie or there are multiple tied teams where a pairwise winner does not emerge, then the team that has the maximum total match score (margin of victory) will be declared the winner. In a rare event two teams have the same score then the team that had the best performance playing as chaos will be used to tie break. If that also can't decide we will toss.

- If a game is tied in knockout round level with same margins then teams will compete against random players from round robin stage in a sudden death fashion. I.e., both teams in tie will individually compete against a random player. Whichever team performs better against the latest random player will win. In case of tie, a new random player will be tried. Even after 5 random players if no decision is arrived, then a toss will be conducted.

What is being provided:

The assignment packet is the same as in Phase 1. The board size is communicated to the client at the beginning of the game as it was before. The code is to be downloaded from the "**default"** branch of the repository: https://bitbucket.org/donraj/entropy

Any updates on the repo for Bug fixes/enhancements shall be notified via piazza. You are encouraged to find bugs, and suggest improvements.

- For changing board size/other server config: see **server/config.txt.**
- It can also be done via the command line args while spawning the server.

Evaluation Scheme

1. We will run three tournaments. The allocated time will be 1 minute, 2 minute and 3 minutes per player per game in a set in the three tournaments respectively.
2. We will create a league fixture per game. In each league there will be many small groups of size about 5-6 teams. Each group will play a round robin and the top team will move to pre-quarter finals. After that it will be a standard tournament with pre-quarter final, quarter final, semi-final and final.
3. Each tournament performance is worth 5 points. All teams which do not timeout and don't send invalid moves will get a score of 1.5 per tournament. Their performance in their group will determine the next bin score which can be a maximum of 2. All teams losing in pre-quarter final will get a total score of 3.5. All teams losing in quarter final will get 4. All teams losing in semi-final will get 4.5. The runners up team will get a 5. The winner will get a 15 and won't participate in the next league (giving a chance to two other teams to get a 15).

Code: Your code must compile and run on **machine named 'todi' or any machine with similar configuration present in GCL.** Please supply a run.sh script for spawning an instance of your player, also supply a compile.sh script for compilation – may it be blank.

The scripts (complie.sh and run.sh) should be in the client folder. Tampering with other part of server code is not recommended, as the autograder script extracts only your client code.

What to submit?

1. Submit your code for your game player = "contents of the client folder ONLY". **The code should be contained in zip file named in the format <EntryNo>.zip.b64** If there are two members in your team it should be called <EntryNo1>_<EntryNo2>.zip.b64 Make sure that when we unzip the following files are produced:

   compile.sh

   run.sh

   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)

Code verification before submission: 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 A2).

What is allowed? What is not?

1. **THE LATE SUBMISSION OF THIS ASSIGNMENT IS NOT ALLOWED.**
2. You must use either the same partner as in Phase 1. Or you may work alone. In case your partner has withdrawn or has audited (and does not want to submit the assignment) you can find another such partner, but you must make sure that your code is not caught in plagiarism against other codes in this assignment.
3. 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.
4. 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 related problem.
5. It is preferable to develop your algorithm using your own efforts. However, we will not stop you from google searching.
6. 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.
7. You get a zero if your player does not match with the interaction guidelines in this document.
8. We will run plagiarism detection software. Any team found guilty will be awarded a suitable penalty as per IIT rules.