ALLCONNECT GAMES (Phase 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.

The Game of Connect Four: In the game of Connect Four two players compete to make a consecutive series of four of the same color. The board is a vertical 6x7(6 rows and 7 columns) board. The players alternate turns and anyone who makes the first sequence of four wins.

The Game of AllConnect(n,m,k): We will instead play a generalization of the game. In AllConnect(n,m,k) our board is a vertical board of dimensions nxm. The players alternate as usual. However, the goal is to make a consecutive sequence of k of the same color. A final modification is that the game does *not* finish after the first k-sequence. Instead the game is carried till the end and finally all sequences of length k are counted for both players. Whoever made more sequences wins by that much margin. Scoring the following configuration in game of AllConnect(6,7,4) will give the player X with score: 3 (three sequences at top row) and player O with score: 6 (one sequence at last column, two sequences at second-top row and three sequences diagonally).

X X X X X X O

X X O O O O **O**

O O X X O O **O**

X X O O X X **O**

O O O X X O X

X O X O X O O

Algorithm: Implement this game as an instance of minimax search with cutoff and alpha-beta pruning. Learn a rudimentary evaluation function (define features and set weights of features either by hand or by a learning procedure pitting your player against yourself). You are encouraged to gain insight in your player by pitting it against other teams.

What is being provided:

Your assignment packet contains code for the game server and sample code for your player in C++ that (1) interacts with the game server, (2) gets adversary's move, (3) generates a random move and (4) outputs the random move to game server.

Interaction with Game Server:

The server compilation that you would be given would be able to run as ./server <portno> <total_time_for_each_player> <N> <M> <K> <mode>. After this you can do a "sudo netstat -ntlp" to check if the server is listening on the port you have provided. Then you can run the first client using ./client <ip_of_server>

<port_no> (for localhost you need to put in 127.0.0.1). Following which you need to start the second client in the similar manner. The server then sends initialization information to both the clients and the client that connects first becomes the first player (The one you start first). After that the clients and the server will exchange the game information. You will only need to fill in the nextMove() function provided in the client's code. This function will make use of the board state that in the random bot is stored in vector<string> board. nextMove() function returns the column number (0-indexed from the left) for which you want to place the next move. Note you might not want to play into a column number >= M (NO_OF_COLUMNS) or else you may lose the game. Also you may not want to play in a column that is already full (else you will lose the game). There is a total time assigned to you that gets decremented when your turn is going on (else you will lose the game). You might not want to exhaust the same.

Finally you can write any auxillary code that you want in the client that we are providing just make sure you maintain a somewhat similar protocol (as there is present in the random bot) to interact with the server.

The mode argument with server lets you run on a bot vs bot mode or a human vs bot mode. Mode = 0 is the bot vs bot mode(explained above). Mode 1 allows you to play in the other mode in which you will have to play when your turn comes by entering a column number at the terminal where you run the server.

Code: Your code must compile and run on **machine named 'todi' or any machine with similar configuration present in GCL.** Please supply a compile.sh script for compilation. Also supply a shell script run.sh. Executing the command ./run.sh server port should start your player and start interacting with game server.

What to submit?

1. 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
2. Submit at-most 1 page writeup (10 pt font) describing your choices and rationale for your player. This is not graded but failure to submit a satisfactory writeup will incur negative penalty of 20% of total score. Your writeup will help us identify any common misconceptions and particularly good ideas for discussion in the class.

Evaluation Criteria

1. In Phase 1 we will test your code against a simple baseline player on AllConnect(8,9,5). Your final score will be based on your player's score – baseline player's score.
2. Extra credit may be awarded to standout performers.
3. The Phase 1 and Phase 2 of the project jointly carry weight of two programming assignments. Phase 1 carries only 25% of this weight and Phase 2 carries 75%.

What is allowed? What is not?

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 the much more open ended final project (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 the partner from assignment 1.

2. You are required to work in C++ for fair comparison amongst all teams. You may choose to not use the sample code.

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 related problem.

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

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.