Group 1: Tyler Brinkley, Susan Foster, Alexander Neumyer, Michael Tang, Melinda Yan, and Julie Yang

**Group 1 - Cubic Tic-tac-toe Alpha-Beta Search Project Proposal**

1: The project's goal is to apply the alpha-beta pruning variant of the minimax game tree search algorithm (shortened to alpha-beta search) and analyze its results for our own innovative version of the classic game, Tic-tac-toe. We will design and implement a cubic version of Tic-tac-toe with its own unique set of rules (rules at the bottom). The analysis portion would consist of comparing our own heuristics used in alpha-beta search with other distinguished heuristics that were designed for the original game but adapted to our version.

2: We will investigate whether our cubic version of Tic-tac-toe can be solved as the classic game of Tic-tac-toe is known to be solved. In the event that we conclude that our version cannot be solved, we aim to find the best set of heuristics.

3: Motivations for this project stem from the classic game of Tic-tac-toe being solved. Small variations of the classic game such as using a 4x4 or 5x5 board are already known to be solved because of the small solution space. However, using a 3D board (specifically 3x3x3) is vastly different from the classic game and thus, presents a curious opportunity to investigate if it is solvable.

4: The game Tic-tac-toe has many admissible heuristics that are already existing. We want to know if those heuristics can still be applied for our 3D version of the game. We also want to know if adapting the heuristics to our new version of the game or creating our own heuristics would result in a higher win percentage and/or more efficient wins. To compare the efficiency of the heuristics we would have to know how fast the players can win the game or how many steps it takes to win.

5: We plan on using a python environment for its ease of use and object-oriented nature. Python’s dynamic nature will also help in the inevitable case of the alpha-beta search yielding potentially large integer values, and would help avoid integer overflows.

6: We will implement our variant cubic Tic-tac-toe in one program so it can be interfaced by both human players and computer players (Bot). At the same time, we will be implementing a random-move Bot, an alpha-beta search Bot that we create the heuristics for, and an alpha-beta search Bot adapted from distinguished heuristics for the classic version of Tic-tac-toe. We will pit the alpha-beta search Bot against the other two Bots multiple times, repeating the matches as we improve the heuristic algorithm that the alpha-beta search Bot employs. At certain intervals in the development of the alpha-beta search Bot, we will also pit this Bot against human players, some without experience playing this game, some with experience.

7: To determine the success of this project, we will analyze the results of the matches between the alpha-beta search Bot and its opponents (random-move Bot or human players). A full success for this project would be to create an algorithm that can beat human players consistently, but in failing that, a partial success would still exist if the algorithm could consistently beat a random-move Bot.

Cubic Tic-Tac-Toe Game Rules

1. Each player goes one at a time placing X’s or O’s like how original tic-tac-toe is played
2. If a player places a move on an edge square, any (directly) attaching edges (on other faces of the cube) are also counted as that player’s move (unless the attaching edge is part of a face that one player has already won).
3. Each face of the cube is like a separate tic-tac-toe game. Winning a tic-tac-toe game on that face means winning that face of the cube.
4. The goal is to win as many faces of the cube as possible.

Diagram, engineering drawing

Description automatically generated