MP2 Report
CS165A

Norman Chung
5550926
normanchung@ucsb.edu

Architecture: Brief explanation of your code architecture (describe the classes and basic functionality)

For my code architecture, I have multiple functions, each with their own purpose. A board initializer to set up the board given the input size while making sure the input size is valid along with who goes first, a move piece function to add a new piece to the board, a checking function to check if the piece added is already taken, a checking winner function used to determine whether the piece put down makes a win in the game or not which checks in the horizontal, vertical, and both diagonal directions, to check when a player has won the game, a checking score function, a diagonal function used to deal with any diagonal patterns, a evalhuer function used for the evaluation/heuristic function, and the minimax function which implements alpha beta pruning.

Search: How you search for the best move (which algorithm you use, what heuristics, optimizations, etc)

For the search, we first do a check on the possible score for each potential move to come, given the most recent input. Here, we check all possible and valid moves, and assign a certain value to it, the score. We then check to see if there are any possible winning patterns observed, and if so, we add onto its score. This emphasizes that this move is better, as it has a higher chance of leading to a possible win. This allows us to have a better representation of the score value, as a move that is very far away from everything might be a possible move, but a move closer to my pieces would have a potentially higher score given a possible win pattern shown. Then, we evaluate these scores, and find the one that gives us the best possible outcome. Whatever has the highest value is the move that the program decides gives us the highest possible chance of winning.

Challenges: The challenges you faced and how you solved them

The challenges I faced were difficult to figure out for a while. Again, I had issues with CSIL dying on me, where I couldn't access my files for a few hours. It was my mistake to not push it onto github or save a copy, but this wasn't something I expected to happen. Other challenges I faced were with the binary file, as I constantly got permission denied errors, and could not find directory errors, which I was unable to fix. Because of this, I was only able to test against a human player, as something was wrong with the light and dark directories, even though I set up everything right, with the mp2 directory inside of it, and the gobang.sh file labeled as gobang and and linux style newline endings. In terms of challenges in the code, the whole thing was a challenge. I didn't realize how hard it was to code a game, and it took me a while to realize all

Norman Chung
5550926
normanchung@ucsb.edu

the possible errors that could occur. Implementing the minimax and alpha beta pruning wasn't too hard, but it definitely took a while to fully understand and put into code, as I had to try multiple times to get the correct function. Setting up the game was rather easy, but when it came to checking everything, it was very tedious, to check for everything.

Weaknesses: Weaknesses in your method (you cannot say the method is without flaws) and suggest ways to overcome them.

The weaknesses in my method are as follows. My code implements the minimax and alpha beta pruning, but I am not completely sure if it is accurate. I followed everything that was taught to us, along with some youtube videos, but I did not make the cutoff for 7/10 games, so I am definitely relying on the 50/60 points with the correctly implemented minimax algorithm, and the alpha beta pruning. A lot of my code was hardcoded, as checking for all possible patterns and such was kind of hard for me. If I had more time and brainpower, I might've been able to simplify the code, in the sense that I could have found some sort of pattern in the patterns, to make the checking function a little simpler.