CS 165A MP2 Report

Nuan Wen (nuanwen@ucsb.edu, 8710790)

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

   I implemented minimax with alpha-beta-pruning. Due to the poorness of my design of heuristic function, I am able to set MAX_DEPTH to 10 and somehow it still runs fast.

2. Search: How you search for the best move (which algorithm you use, what heuristics, optimizations, etc)
   The heuristic evaluations I came up with are centered around 4 areas as in the comment of my heuristic function:
   0. centerness: occupy the center gives some bonus
   1. liberty: like the concept in go
   2. vulnerability: check if it is under attack
   3. completeness: check if it is connecting the right sides
   And I implemented each of them separately and add all the scores together.

3. Challenges: The challenges you faced and how you solved them
   I had three challenges.
   The first one is trivial, yet deadly. It is about the incompatibility between my windows machine and CSIL. I was receiving tons of error messages on indentation and neither vim or emacs (access remotely) or atom on CSIL is able to quickly fix such problems. Later on I realize in Visual Studio there is a plug-in that can detect invisible indentation problems and I was saved.

   The second one is about my minimax algorithm. It is hard to tell why the performance is bad. Initially I believed it is just because of my poor heuristics and I just spent most time trying to come up with better heuristic. However, it turned out my minimax is buggy: I forgot to retract moves!

   The third one, as I mentioned earlier, is about the heuristic function. Given the time that I can work on this project, I do not think I am able to write an efficient shortest-best-path-search algorithm, so I decided to only provide some general guidance. The performance on small board does improves, but on larger ones it is still very bad.

4. Weaknesses: Weaknesses in your method (you cannot say the method is without flaws) and suggest ways to overcome them.
   The most obvious weakness is that I did not implement a search algorithm that can find a path to win. Given more time, I will definitely work on that. Also, I think my heuristic function is too focused on smaller boards (like 3*3) because of how I do the testing. Next time, I will avoid doing so.