P3 Write-Up
P3 Due Date: Monday August 19, 11:59 PM

Only ONE person from your group should submit the write-up to Gradescope. Please make sure that you select corresponding pages for each question when you are submitting your write-up to Gradescope. You should also add your partner to your group on Gradescope after you have submitted your write-up.

Project Feedback

(1) Project Experience
Answer the following questions about your experience doing the project.

- How was your partnership? What worked well? What would you do differently next time?
- What was your favorite part of the project? What was your least favorite part of the project?
- Did you enjoy the project? Why or why not?
- How could the project be improved?

Experiments

(2) Node Estimation
Hypotheses: Suppose your minimax bot goes 3-ply deep. How many game tree nodes do you think it explores? (we’re looking for an order of magnitude)

Results: Run an experiment to determine the actual answers for the above. To run the experiment, do the following:

- Run SimpleSearcher in a game and capture the board states (fens). To do this, you’ll want to use code similar to the code in the testing folder.
• Now that you have a list of fens, you can run each bot on each of them sequentially. You’ll want to slightly edit your algorithm to record the number of nodes you visit along the way.

• Run the same experiment for 1, 2, 3, 4, and 5 ply with both implementations (use ply/2 for the cut-off for the parallel implementation). Make a pretty graph of your results and fill in the table in the write-up template as well.

Conclusions: How close were your estimates to the actual values? Did you find any entry in the table surprising?

(3) Optimizing Experiments

THE EXPERIMENTS IN THIS SECTION WILL TAKE A LONG TIME TO RUN.

NOTE: Because chess games are very different at the beginning, middle, and end, you should choose the starting board, a board around the middle of a game, and a board about 5 moves from the end of the game. The exact boards you choose don’t matter (although, you shouldn’t choose a board already in checkmate), but they should be different.

Sequential Cutoffs

Experimentally determine the best sequential cut-off for your parallel search. You should try to test this at depth 5.

Plot your results and discuss which cut-offs work the best on each of your three boards.

Comparing the Algorithms

Now that you have found an optimal cut-off, you should compare the actual run times of your two implementations. At depth 5, using your optimal cut-offs, time both of your algorithms for each of the three boards.

Plot your results and discuss anything surprising about your results.
(4) Beating Traffic

Traffic

In the last part of the project, you made a very small modification to your bot to solve a new problem. We’d like you to think a bit more about the formalization of the traffic problem as a graph in this question.

- To use Minimax to solve this problem, we had to represent it as a game. In particular, the “states” of the game were “stretches of road” and the valid moves were choices of other adjacent “stretches of road”. The traffic and distance were factored in using the evaluation function. **If you wanted to use Dijkstra’s Algorithm to solve this problem instead of Minimax, how would you formulate it as a graph?**

- These two algorithms DO NOT optimize for the same thing. (If they did, Dijkstra’s is always faster; so, there would be no reason to ever use Minimax.) **Describe the pros and cons of each of the algorithms. When will they output different paths?**

Above and Beyond

(5) Extra Credit

If you did implement Alpha-Beta and Jamboree searchers, include analysis of them in the (2) and (3) experiments you performed above.