Report

1 Title
Baroque Chess Agent

2 Team member names
Jingqiang Wang, Junhao Zeng

3 Intended personality of the agent
Thanos, a bad guy who is aggressive.

4 status of agent

4.1 features that work
Have feature for scanning the board in row-major order (needed for parameterized_minimax);
Have feature for Generating all possible non-capturing moves for any Friendly piece at the current square;
Have feature for basic static evaluator;
Have feature for minMax search;
Have feature for Alpha-beta pruning;
Have feature for fixing move to avoid moving any frozen piece;
Have feature for adding in the generation of capturing moves for withdrawer, leaper, pincers, kings, coordinator;
Have feature for iterative deepening search in makeMove function

4.2 features that don't work
Don't have feature for imitator captures.

5 design retrospective
The core principle of our agent is using IDDFS from lower depth to higher depth to find the best move by minimax search, which leads to the best state. Due to the time limit, it will return the optimal movement of the last layer that is searched completely as our decision. In the minimax search, our agent will keep searching the successors of the first state and updating the optimal state value as well as the move leads to that state. During the search, the agent will record the alpha and beta value and prune the states which are not worth to search. As for evaluating the state, we give each live piece a special value based on how powerful this piece is and sum them up as the value of this state. 'N_STATES_EXPANDED' and other required value are recorded during the search as global value.
For the movement part, there are eight moving direction for each piece except for Pincer. So for each direction, how many steps a piece can move forward, how many potential states there are. Besides, before my agent moves any piece, it will check whether the piece is frozen by the enemy freezer, If the piece was frozen by enemy freezer, my agent would not move the piece.
For the capture part, my agent have all the capture rule except for imitator. The general capture rule is that when my agent moves a piece to a new square, it will check whether the moving piece can eliminate some enemy based on its specific rule.
With the movement part and the capture part, my agent can generate a lot of possible states, then I use static evaluation, minMax search, alpha beta pruning and iterative deepening to get a good move.

6 optional partnership retrospective

6.1 how we operate as a team
First, we both read the homework specification, and list the works that need to be done. After we listed the work, we find out there are two major part for this homework, one is the AI

algorithm part, one is the baroque chess rule part. Therefore, we decide one person writes the AI algorithm part, and one person writes the baroque chess part.
Besides, since the AI algorithm part needs successors of board to debug, So the one responsible for baroque chess part starts earlier to develop a basic baroque chess rule which only has movements, so the other person can start to write AI algorithm part.

6.2 what issues you had in terms of collaboration
Some bugs seem to involve two parts of the code, the AI algorithm part and the baroque chess part. Since each of us only writes part of the code, so some bugs are hard to solve.

6.3 how we overcame those issues
We work together to locate the bug first, and then let one person to solve the bug depending on the location of the bug. If one person can’t solve the bug individually, then two people work together to solve the bug, no matter where the bug is.