Adversarial Game Playing Agent Analysis

About the Knights Isolation

In the game Isolation, two players each control their own single token and alternate taking turns moving the token from one cell to another on a rectangular grid. Whenever a token occupies a cell, that cell becomes blocked for the remainder of the game. An open cell available for a token to move into is called a "liberty". The first player with no remaining liberties for their token loses the game, and their opponent is declared the winner.

In knights Isolation, tokens can move to any open cell that is 2-rows and 1-column or 2-columns and 1-row away from their current position on the board. On a blank board, this means that tokens have at most eight liberties surrounding their current location. Token movement is blocked at the edges of the board (the board does not wrap around the edges), however, tokens can "jump" blocked or occupied spaces (just like a knight in chess).

About The Project

In this project, agents have a fixed time limit (150 milliseconds by default) to search for the best move and respond. The search will be automatically cut off after the time limit expires, and the active agent will forfeit the game if it has not chosen a move.

The agent is evaluated by running games against another opponent agent and counting the percentage of wins by the agent.

The opponent agent uses a random choice for moves in the first 2 turns or plies . Beginning with the 3rd ply, the opponent agent uses a Minimax search algorithm to choose the move it will take. The scoring heuristic used by the opponent will be discussed later, as it will be used as a baseline heuristic to compare with a custom heuristic used in the final testing.

About the Agent

A number of potential improvements have been implemented and evaluated to improve performance over a baseline agent such as the opponent agent described above. The Minimax algorithm was used, and Alpha- Beta pruning was also used. Both were compared in the final test.

Iterative deepening up to a possible depth of 10 plies was carried out in both the Minimax search and the Minimax with Alpha-Beta pruning search. In the final test, the searches with iterative deepening were compared to those without which one search was used at a depth of 3 plies, similar to the baseline agent of the opponent.

The Minimax search algorithm uses a heuristic scoring function to evaluate the game status and select an action from the available possible actions when the maximum depth is reached. Compared to a baseline heuristic, a custom heuristic has been implemented and evaluated. The heuristic baseline was the same as that used by the opponent, the number of moves(freedoms) available to the opponent minus the number of moves available to the opponent.

Baseline Heuristic: N. of agent moves/ N. opponent moves

The custom heuristic used both of those number of moves, but also utilized information about how far each agent was from the center of the board.

Custom heuristic:
((opponent distance to center + 1) * (number of agent moves)) -
((agent distance to center + 1) * (number of opponent moves))

Then the various options discussed above were tested. In each case, the tests conducted 1000 games and the winning percentages for the agent are listed in the table below.

	Minimax				Minimax w/ Alpha-Beta			
Iterative Depeening	No				Yes			
Opening Book	No	Yes	No	Yes	No	Yes	No	Yes
Baseline Heuristic	50%	50%	61.7%	64.1%	50%	50%	62.4%	64.7%
Custom Heuristic	53.6%	51.7%	69.7%	70.7%	54.2%	55.7%	69.1%	72%

Custom Heuristic Questions

1. What features of the game does your heuristic incorporate, and why do you think those features matter in evaluating states during search?

As discussed above, the custom heuristic includes two features of each agent, the number of movements / freedoms the agent has and the distance between the agent and the center of the

board. The number of moves that the agent has is important because the game ends and the agent loses if the number for an agent becomes zero. The distance to the center was chosen because in future plies, a location in the center generally gives states with more potential movements. Generally speaking, states far from the center have fewer moves because agents can not move outside the grid.

2. Analyze the search depth your agent achieves using your custom heuristic. Does search speed matter more or less than accuracy to the performance of your heuristic?

The baseline and custom heuristics would go to the maximum depth with a similar frequency in the iterative deepening tests when testing the agent. Both heuristics are calculated with a similar subtraction, and each time the score was calculated, the distance to the center of the board was accessed using a search table. For this reason, the search speed does not seem to be the reason for better performance using the custom heuristic. Incorporating the distance from the center is more likely to add more accuracy.