Artificial Intelligence - Project 1

Students

Ermir Pellumbi

Wojciech Woźniak

Yann Le Lorier

Elías Friðberg Guðjohnsen

Model of the environment

State of the environment

- Width
- Height
- For each cell in the environment - what is on it (black - 2, white - 1 or nothing - 0)

Successor state for an action

After an action occurs, the state generated only changes the cells in the environment that were affected, meaning where the piece comes from and where is is going, and if another piece was captured.

Legal actions

The legal actions for any state can be separated into the following cases:

Case 1: There is nothing in front of the pawn and no enemies at the front diagonals

> The Player can move only one cell forward

Case 2: There is a piece in front of the player

> The player cannot move

Case 3: There are enemies on the front diagonals and there is nothing in front of the pawn

> The player can choose between going forwards, or a front diagonal move

Case 4: There is something in front of the player and enemies on either front diagonal

> The player can only attack either front diagonal

Report

We ran the experiments principally on the small gdl files (5x5 and 3x5) with no times constraints at first. we focused primarily on getting the logic of the main algorithm correct before jumping into technicalities of the game (time constraints, legalities of the moves) while developing those on the side.

Evaluation function

In our case we evaluate only the number of pawns. We count the number of white and black pawns and then subtract these values and return it as a evaluation value.
We also tried to consider the location of the pawn that is closest to the final stage. If the pawn is closest to the final state, the more points we get in the evaluation.

Results (playing against random opponent)

Size of the Board	Opponent	Win	Draw	Lose	# of Runs
5x5	Black	7	0	2	9
3x5	Black	5	3	1	9
9x9	Black	4	0	0	4
5x5	White	7	2	0	9
3x5	White	4	4	1	9
9x9	White	4	0	1	5

We took most of the experiments at 3x5 board with time constrain that was 10 seconds. During tests we did both, we played against our agent and we let the random agent to play versus it. When we played against the agent, we never managed to lose. The most common result was a draw, but we managed to win a couple of times against it. While our agent played with the random agent, the results were more randomized. The most common result was also a draw, but there were also some loses and wins.

Improvements made

Prioritize capturing moves:

While checking for the legal moves, the capturing moves has priority over the forward move. The reason for this is that a capturing move will have an impact on the evaluation function, since the number of opponent pawns will decrease after such move, along with the fact that the probability that it is a better move than a forward move is high.

Legal moves:

Checks the legal moves from the opposite side, all the way to the agent's side. *i.e.* when playing as white, it checks the legal moves from bottom to top, and when playing as black, from top to bottom.

Interpretation of the results

The heuristics we took isn't perfect in every cases. It has a priority to capture other pieces, instead of analyzing the positions of pawns and going to the final stage.

The reason why our agent does better in bigger environments against a random opponent, is because in a smaller environment, the probability of making a good move with random heuristic is bigger, and the closer we get to a bigger board, the better it is to have some kind of heuristic that is not random.