

Chess Project

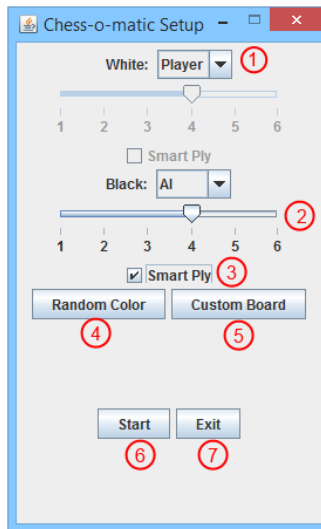
Adam Balint
5141619
COSC 3P71
2016/01/06

Description

A functional chess program using minimax with alpha beta pruning.

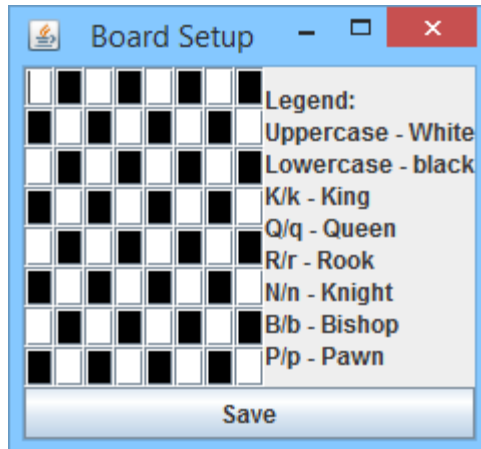
How to use

Setup:



- 1: Select between an AI or Human Player
- 2: Select the ply depth used by the AI
- 3: Select if smart ply is used.
 - Effects: if less then, 12 pieces are on the board, the ply is increased by 1
- 4: Randomize who takes black and who takes white
- 5: Generate a custom starting board
- 6: Start the game of chess
- 7: Exit the program

Custom Board:



Description: Each of the black and white squares are a text field. Type in the pieces in the desired position according to the legend. Click save and then press the start game button on the setup to start the game with the saved board. If it is decided that a custom board is not needed, then the board setup can be cancelled by closing the window.

Note: The board will only be used one time.





Note 2: The black pieces always go at the top, and the white always at the bottom

In the Game:



- 1: The player colour and whether it is a person (P) or an AI (A)
- 2: The turn indicator - 4 different versions described below
- 3: The container that holds the captured pieces
- 4: The game board where moves can be made and viewed

Turn Indicators

Icon	Meaning	Obtained by
	Normal turn	None of the below happening
	Check	The king being attacked by the enemy player
	Checkmate	The king being attacked by the enemy player without any move to defend. This is a game end state
	Stalemate	The game having no moves that can achieve victory for either player. Example: - each side has only the king

Making a Move:

In order to make a move, select a piece. Once a piece is selected, the available moves will be shown in green. Next, select one of the available moves to move, or select a different piece to move that piece. Selecting a move will cause the piece to move, and will end the turn of the current player as shown in the figure below. Note the indicator icon has been changed to show that it is the black player's turn.



Piece Selected



Piece Moved

About the Program

Board Representation:

In this program, the board representation in the code differs from the board representation used in the custom board option. In the code, pieces are represented by the positive and negative versions of the numbers 1, 2, 3, 4, 5, 6 for pawn, bishop, knight, rook, queen and king respectively. Positives represent white pieces and negatives represent black pieces.

Move Generation:

Most moves are calculated by adding all of the theoretically valid moves to the list of moves that a piece can make. A check is done after all of the moves are added to see if they are valid. A valid move is a move that does not leave the king in check or blocks check if the king is in check, and is on the board. If a move ends on a square that is occupied by a piece of the same colour, then that move is also considered invalid.

The AI:

The AI uses a minimax search with alpha-beta pruning. When making a move, the algorithm keeps track of what it has done to the board, and it reverses each move back to the board's original state before checking the next move. When a leaf node is reached (ie. the ply depth is exhausted), or if there are no moves, a heuristic is used to get the value of the current board which is then returned.

The Heuristic:

The heuristic takes into account multiple factors. The most major component is the difference in piece values. This is calculated by taking the difference of the number of pieces between the two sides for each piece, and multiplying it by the value of the piece. The difference in mobility between the two sides is also considered. This is calculated by taking the difference in size between the number of unique moves on each side and multiplying the result by a small value. The third factor considered concerns the different states of the game. A check is assigned a small bonus as having the opponent's king in check is beneficial. A checkmate provides a large increase in score, while a stalemate is assigned a score of 0. The final part of the heuristic deals with castling. This part specifies that if the king has moves, then there will be a small decrease in score. A similar idea with each rook, however there is less of a deduction as the king may still castle on the other side if a rook is moves. The castling portion of the heuristic was added in to help prevent the AI from making useless moves by moving the rook between two adjacent squares.

Known Problems

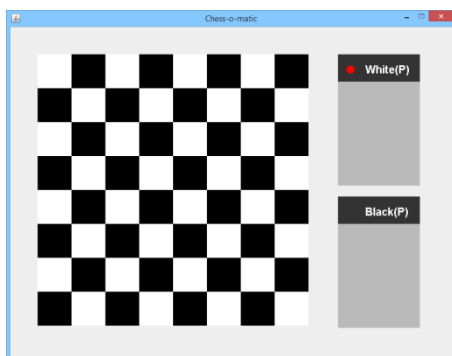
A known problem is that when two AIs are playing against each other at the same ply level, then they are likely to get stuck in a move loop. Another known problem is that when the player has only a few pieces, the AI sometimes has difficulty getting the player in checkmate.

Future Additions

- Fix the repeating moves for AI
- Improve Heuristic
- Speed up algorithm

Troubleshooting

Blank Board:



If the board is displayed as shown in the picture, then the jar file is not in the same directory as the images folder. To fix, place the jar into the folder containing the images folder (shown below) and then run the jar again.

Name	Date modified	Type	Size
Images	2016.01.06, 3:11	File folder	
Chess Project.jar	2016.01.06, 2:40	Executable Jar File	33 KB

References

Research:

<https://chessprogramming.wikispaces.com/>
https://en.wikipedia.org/wiki/Computer_chess
<https://en.wikipedia.org/wiki/Chess>
Dr. Ombuki - COSC 3P71 Lecture Slides (2015)

Images:

Piece Images:

https://commons.wikimedia.org/wiki/Category:PNG_chess_pieces/Standard_transparent

Wood Background:

<http://www.freelargeimages.com/wood-texture-1462/>

Marble Texture:

<http://bgfons.com/download/4665>

Crosshair:

https://commons.wikimedia.org/wiki/File:Crosshairs_Red.svg

Cartoon Skull:

<http://photobucket.com/images/skull%20pixel>

Hourglass:

http://piq.codeus.net/static/media/userpics/piq_22602_400x400.png