Chess Game Player

Name: Siddharth Gupta
Mentor: Prakhar Banga

Introduction

Chess is one of the most played games around the world. Since the advent of computers, it has been a quest for all programmers to create a program to beat the human mind at chess. And it was after 50 years of thorough research that success was finally achieved. A program was actually made to that could defeat the world's top chess player. Chess programs are hard to make since they have to be incorporated with a human like intelligence, i.e., Artificial Intelligence or A.I. as it is called. And to embed human like intelligence and reasoning in a computer program is a really tricky job.

But the part where the computers gain an extra edge is their capacity to calculate humungously at an eye-blinking speed, i.e. Computers can process information really quick. Thus, this capacity makes a brute force sort of method possible to compute all possible variations of a certain game of chess.

Motivation

Games of chess are, no doubt, very interesting. But to find an opponent to play with whenever one wants to play is a small but genuine problem. Its not necessary that others are not working at the same time when you are not, or alternatively they also desire to play a game at the same very moment. But having a computer playing chess rules out all these problems. You can play the game whenever you want, at whatever difficulty you desire, and even analyse your entire game later. There is a huge set of possibilities with computer playing chess, and a tremendous scope for improving your playing skills.

Details

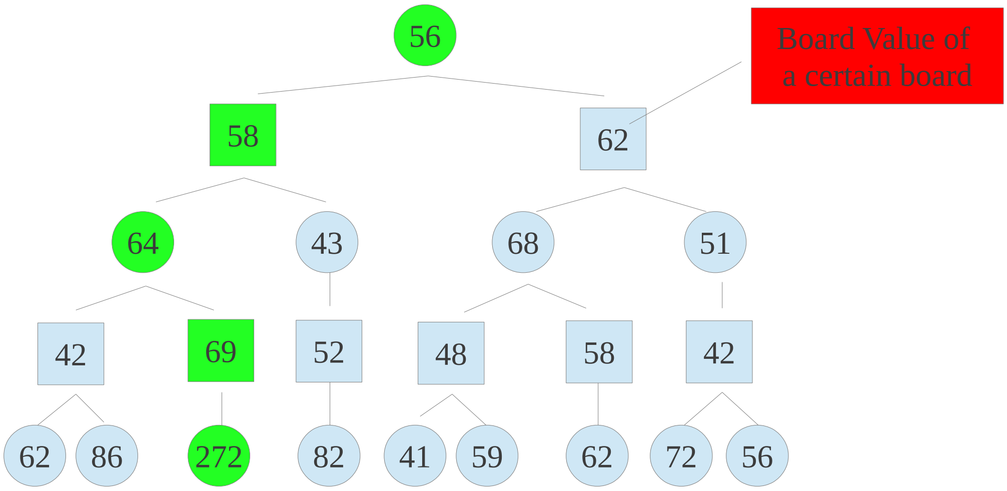
“Chess Game Player” as developed by me plays using a brute force approach. It lists out all possible moves in a tree like structure upto a certain depth and then tries to choose the best possible sequence of moves. It has a set pattern of opening moves which form a defensive castle like structure to gain an extra edge. If any of the opponent's piece cross the 4th rank, then the control is transferred completely to the computer and then it starts computing moves using the tree algorithm. The level of difficulty can also be adjusted in the game. The difficulty level is dependent upon the depth upto which the tree is made, but as the depth increases, the computation time increases exponentially.

Interface

Not much attention has been given to the GUI of the project. The program runs on the terminal itself and the user has to give input from the keyboard, mentioning the positions which are involved in the move, and the corresponding move will be made by the program if its valid, else an “Illegal Move” statement is shown. The program also provides features as saving the game and resuming it afterwards by loading the file. A list of all the moves is also saved in a file “Moves” for later reference and analysis.

Implementation

The program is based on the minimax algorithm. On computer's move, the minimax function is called, and it computes all possible moves it can do (thus creating new boards), and for each subsequent board, it further branches all possible boards for the opponent (the user) and so on upto a certain depth. For each board that is generated, the value of the board is found based on the positions of different pieces and moves available to them. The values for the computer's piece are positive and those for the user are negative, thus the user tries to reduce the board value, while the computer tries to increase it. The branch that results in maximum value is selected.



Once the best move is decided, changes are made on the board and the board is printed. When the user enters the move, the program checks whether the entered move is legal or not, and accordingly either makes the move or declares it an illegal move.

The program also allows some special moves like “Castling” and “Promotion of Pawn” and all the conditions necessary for these moves are first checked and then only they are allowed to be done. In the opening of the game, a fixed set of moves are made by the computer to create a defensive Castle like structure. These fixed set of moves are abandoned if one of user's piece crosses the 4th rank on the board and the control is transferred to the minimax algorithm. Else these moves are followed until the castle is complete.

Also, the king is allowed to move only to those places where it is not under direct attack by the opponent. This is done by scanning the entire board and checking if any piece has access to the place where the king is about to move. The same is done to check if its mate for either player. All the places around the king are checked along with the place where the king is to see if any of opponent's piece can attack them. Not only this, it is also checked if it is possible for any of the pieces of the player, who is under check, to block the path of the attacking piece to save the king from check. If all the desired conditions evaluate to true, then only mate is declared.

Deliverables

The program obviously is not able to defeat human players with small depths but certain level of "nice gameplay" can be seen if the depth of the tree is increased, but this takes a lot of time, as by each depth, the number of nodes increase exponentially and so does the time for execution. Thus, it takes computer a lot of time to think each move. The pieces are represented by letters. Black and white are replaced by lower case and upper case letters. Computer's move is also displayed for the convenience of the user when the move is made.

Some Bugs

Like all other programs, the chess player developed in this project also has some bugs. A few that are known are as follows: the computer is not forced to move its king even if its given a check. Instead the game can be ended by capturing the king. Some other rare ones involve some "ghost pieces" (as I call them). The computer starts making weird moves from positions where no piece is present to some other random position and gets stuck with the same move for a long time. This one is a bit less occurring but yet it got caught in one of the test runs, and never occurred in any other run.

Conclusion

Chess has always been a game of high respect due to the amount of mental effort it requires, and since the advent of computers, it had been long sought to induce the ability to think into the computer. With the development in the field of artificial intelligence, it has become possible to code up full-fledged chess programs in computers which can beat even the top human players. And still the advancement continues to seek better and smaller codes for flawless chess play.

References

http://en.wikipedia.org/wiki/Minimax
http://chessprogramming.wikispaces.com/Evaluation