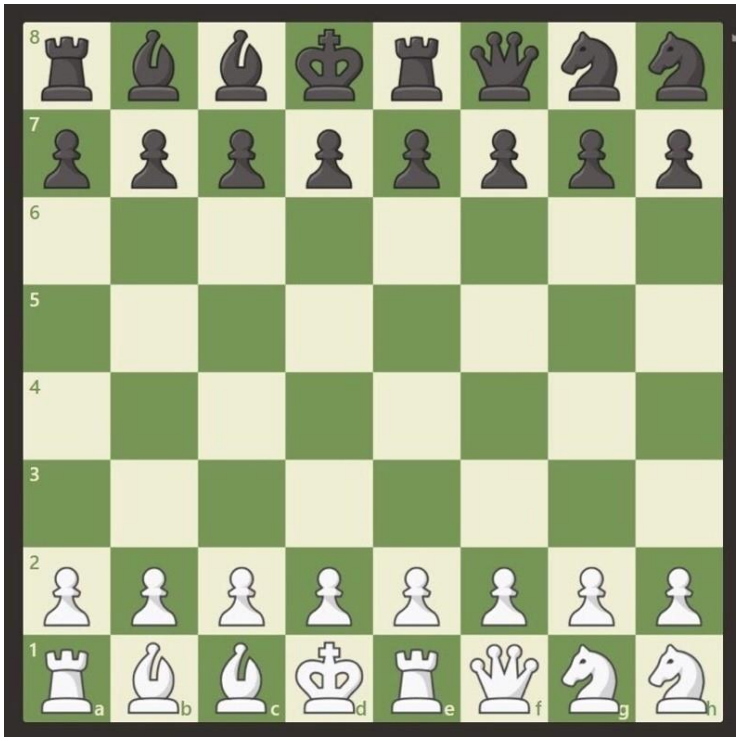
Analysis

Context

Chess is a two-player, turn based, information perfect, strategy board game. Playing chess requires logic, rational thinking and patience. Learning and improving at chess enhances mental prowess and is proven to directly link to academic performance. This comes from forcing the player to adapt to unforeseen scenarios and plan ahead, keeping their opponent's response in mind. However, one aspect of the game reduces this onus and can even remove it entirely. The initial conditions of every game are the same. Whilst there are still many different lines of play at the start of the game, or openings, a select few of them are significantly better than others. Instead of using logical thinking to work out the most effective moves as one does in the rest of the game, it is far more advantageous to learn openings by wrote as this saves on time and possible errors one might make. As a result, fewer strategies are viable, so the start of the game becomes a battle of memory rather than intelligence. Following on from this, players tend to use the opening they are most proficient with meaning the course of the game is sent down a familiar route to them reducing their reliance on intelligence and creativity as they make plays based on experience and memory having often been in similar situations.

In 1996, the world chess champion, Bobby Fischer, invented his own variant of Chess to attempt to force players to rely more on creativity and intelligence rather than experience and memory. He created Fischer Random Chess or Chess960 in which the back rank of the board is randomly shuffled under certain limitations and mirrored on both sides giving 960 possible starting setups. This significantly reduces the player's ability to rely on memory as its infeasible for one to learn openings for 960 different permutations of the game. In 2019 this format was first recognised by FIDE who held a World Fischer Random Tournament. Below is an example of a possible initialisation of Fischer Random Chess.



Bobby Fischer's game is a significant improvement, but it is not perfect. Whilst memorising openings is useless, experience can still play a large factor so someone with more chess experience is likely to defeat someone with more intelligence but less chess playing experience. This is because many aspects of the game are still very similar to those of normal chess e.g. same pieces are used with a relatively similar set up, same size and shape board, same goal. Chess variants are based on chess so the bias towards experienced chess players can never be fully removed but it can massively be reduced.

Objective

The features of my program will facilitate rapid and easy generation of new, user defined chess variants. This should remove the power of memorising and significantly reduce the importance of experience gained from conventional chess, resulting in a level playing field where only creativity and intelligence are conducive to victory. The Objective is to essentially apply the 'Fischer Effect' to an exaggerated extent. This will be done by allowing the user to create custom board sizes, shapes and setups; create custom pieces with user defined movement and select from different rules. This will create scenarios differing massively from those found in regular chess reducing the impact of experience while still requiring rigorous logical thinking. These changes will also change how powerful certain pieces are as their value is based on their movement and the user-defined factors will affect how useful this movement is. As a result, players will have to re-evaluate the power of familiar pieces and strategies based on the new environment they have been placed in.

Chess and other popular strategy board games like Shogi have been optimised and balanced over 1000s of years so they are naturally going to be more balanced and generally more fun than anything a user designs, but if a game can be solved or weakly solved, although this makes it a bad game in long term play (after many games and improvement due to competition), in the short term it is still a very intellectually stimulating challenge and, until a player manages to solve it, provides a good mental proving ground. Once the complexities of the game have been fully explored it can be deleted and a new one can be tested. Similarly, poorly designed variants may get boring faster than chess but once this happens players can just move onto a new variant. In both regards, although chess is probably a better game the variety and mutability of playing variants makes up for these weaknesses.

Why does this benefit from an automated solution?

Firstly, to introduce any sort of randomisation a computer system is much faster than a user generating their own random setup with a dice, for example. Whilst it is possible to play Fischer random chess with a regular chess set, extra equipment is needed for anything involving different pieces and especially for different sized boards. On top of this, it is easy for humans to make mistakes especially with new unfamiliar rules so having an engine which only allows legal moves is very useful. Finally, having a simple automated system allows for much faster design and play of chess variants.

Possible uses/benefits

1) Development of mental prowess

   No memory and little experience are involved so intelligence is always necessary making this a better brain training exercise than chess.

2) Practice environment / Puzzles

The program could be used to create practice environments where movement of pieces and specific scenarios could be tested. Simple, open environments could be created to learn/test the movement capabilities of pieces. Preparing for endgame scenarios that often arise is a very common practice amongst students of chess and such scenarios would be easy to set up and replay with this program. Similarly, chess puzzles are a very good brain training exercise and this engine would be perfect for creating them and using them as practice, especially because many involve fairy chess pieces which don't exist in conventional chess editors but can be easily implemented into this program.

Example of fairy chess puzzle:

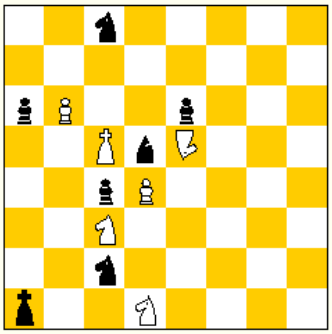
https://www.chessvariants.com/problems.dir/prngtrd1.html

the Nightrider

Fairy chess problem composers offen make use of fairy pieces': pieces different from those that appear in a normal game of chess. One of such fairy pieces - and one which has been used very often - is the Nightrider, a piece that can make more knight-moves in the same directions in one turn.

The Nightrider moves as follows: it moves as a knight, but can continue to move in the same direction with additional knight moves, provided that the leaping points' are all empty. For instance, in the diagram below, the nightrider on e5 can move to c1 (as d3 is empty), but it cannot move to a4.

The problem below was composed by T. R. Dawson; published first in the Problemist in 1927, and appears in Dawsons book Caissa's Wild Roses, which appeared in *Five Classics of Fairy Chess.* The nightrider is displayed as an upside-down knight.



White:
King c5; Nightrider e5; Knight c3, d1; Pawn b6, d4.

Black:
King a1; Knight c2, c8; Bishop d5; Pawn a6, c4, e6.

Mate in two moves.

(Mate in two moves means: find a move for white such that he can mate black after every move of black.)

3) Fun

User is given more creative freedom, not just in playing, but also designing interesting and challenging game modes.

The following are some responses to a question, posted on the r/ChessVariants subreddit, asking people what they enjoyed about chess variants.

9 points · 2 days ago

I just like variety. Chess is great, but there are so many other potential good games out there too.

It's not weird for people to want to play more than one computer game, so I don't think it's too weird for people to want to play more than one type of combinatorial strategy game too. I think it's a shame that most chess players don't want to even bother trying other games of this genre.

5 points · 2 days ago

For the same reason Fischer made 960: openings aren't so thoroughly studied, meaning it's more about thinking on one's feet.

3 points · 2 days ago

If someone always ate the same food, wore the same shoes, ran the same track, sang the same song, or played the same card game, it would be considered weird. People would think that person is boring.

Why this isn't the case with chess is beyond me.

2 points · 2 days ago

I like needlessly complicating things.

2 points · 2 days ago

Because discovery is a greater joy than mastery.

2 points · 2 days ago · *edited 1 day ago*

I'm deeply fascinated by how when you change the rules, different strategies come into being as an emergent property of those rules.

2 points · 2 days ago · *edited 2 days ago*

I think that subjective and qualitative aspects of a game are more important to a person's enjoyment of that game than the mathematical complexity of a game. Let's face it; most people, even GM chess players, have not mastered checkers or draughts, yet they scoff at checkers and draughts as simple games.

Chess is a an excellent game as-is and the tradition/standardization helps with tournaments and international appeal of it. Still, it does have certain issues like many draws even when both players are not that great.

I like Chess960 and while it's true that intermediate players need tactics more than openings, I think: why NOT just eliminate the rigidity of the opening setup?

2 points · 1 day ago

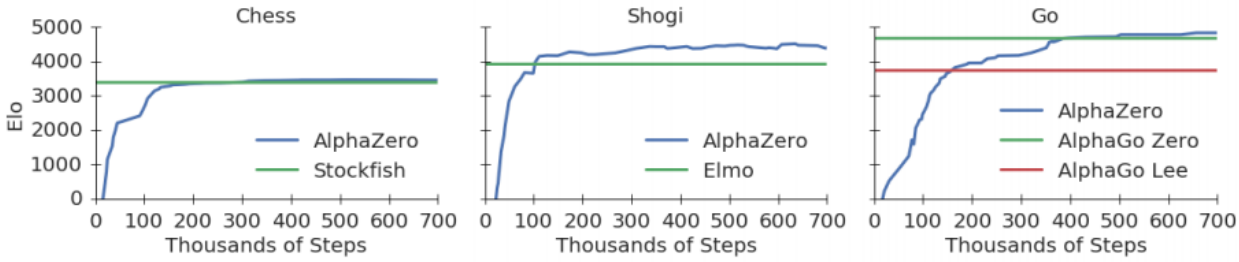
I have no particular reason to be attached to Chess. I am not interested in going to Chess tournaments, and I can play Chess variants against my computer or online with other people. So, I don't need to focus on the game I can find the most opponents for. Chess variants frequently offer the same appeal as Chess with novel twists that add interest. Also, by focusing on Chess variants in general, I can keep my focus on the intellectual challenge of Chess-like games rather than on mastering one particular game that many people are already much better at than I am.

1 point · 1 day ago

I love variety, and I also kind of suck at normal chess.

4) Artificial Intelligence training ground

This program could potentially serve as a training ground for general purpose AI. General purpose AI are normally tested by allowing them to reach their full potential at one task and then seeing how the mental acuity gained from that task translates to other tasks. For example, Open AI's Open AI Five was trained on DOTA 2, a strategy video game, and with this knowledge was able to quickly learn how to dextrously manipulate a robotic hand to solve a Rubik's cube. Like DOTA 2, this program is a complex strategy setting perfect for training general intelligence. It has far more permutations than chess meaning the AI won't get forced into a narrow style of thinking and reach a local maximum limiting its potential. The learning practiced in this program can then be taken and applied to other things. Below are graphs comparing the general-purpose system, Alpha Zero, to narrow AI designed to play specific games.

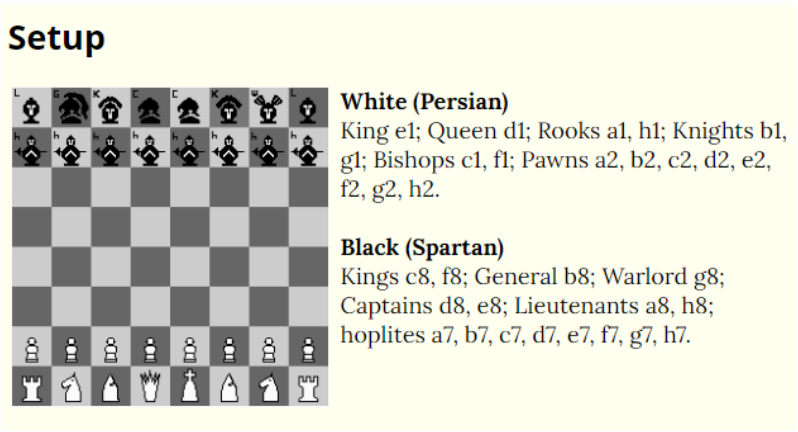


5) Historical Teaching Aid

As well as a game, chess is essentially a simulation of war and in the middle ages it was used to teach military tactics to nobility. One can accurately represent famous historical battles with different variants of chess and emphasise the different armies, strategies and tactics used. Therefore, this program could be used to create abstractions of battles for educational purposes.

An example of this is spartan chess in which the white army mimics the tactics and strategy of the Persians and the black mimics that of the Spartans.

Spartan Chess:



Similar systems

Certain major chess applications such as Chess.com and Lichess allow users to play a list of renowned chess variants. The website Chess Variants Pages contains an extensive list of chess variants explaining their rules in detail. Some of these can even be played in the browser. Users can submit their own chess variants but only once they are moderated and accepted will they eventually be posted on the website. Most applications in the chess variant sphere allow playing of variants but no user defined rules.

There is an independently developed application called ChessCraft which allows limited user defined movement of custom pieces and custom board shapes, but their maximum board size is 7 by 7 which limits movement potential quite significantly.

Proposed solution

Chess and strategy games in general benefit from having simple designs with unnecessary details abstracted. Additionally, chess is turn based and is intended to be played slowly with time for thought between moves. Therefore, I think python with the pygame module is perfect for tackling this problem because it facilitates very easy design of simple games but is less effective for games in real time or with complex graphics. In terms of computational complexity, chess variants are very simple games.

Requirements

1) Text based interface that gives the user 5 options: play default chess; play Fischer random+ chess; create a custom ruleset, delete a custom ruleset, play a custom ruleset.

2) Selecting default chess starts a conventional game of chess without 'en passant' or 'castling' (see omissions).

3) Selecting Fischer Random+ Chess prompts the user to enter a board height and width within the range of 4-26. A board is generated using the user's constraints where the second and penultimate ranks are filled with pawns and the first and final ranks are randomly filled with knights, bishops, rooks, queens as well as one king. The sides are mirrored.

4) Selecting to create a ruleset gives the user the options to design their own pieces (see requirement 5), select from one of the four available win conditions (see requirement 6) and choose between the default board or a custom board of their own creation. The user is then prompted to name their ruleset.

5) Pieces can be defined with 2-dimensional leaping and riding vectors. They can be given any number of leaping and riding vectors the user desires. The status of the can also be set to royal. This means it is involved in win conditions (see requirement 6). Up to 3 pieces can be defined. When a pawn is promoted, the user is offered to select from all the normal pieces, including any custom pieces the user has created.

6) The User can choose from four win-conditions.

1. Checkmate： This is the same as in the default rules of chess. The rules of check, checkmate and stalemate apply to every royal piece.
2. Regicide: rules of check do not apply. The game is won if every enemy royal is captured.
3. King of the Hill: All the rules of checkmate apply as well as hill squares. The game is won if a friendly royal reaches a hill square.
4. Extinction: the game is won when all enemy pieces are captured.

7) The user is offered the option of the default board or a custom board. If a custom board is selected the user is prompted for a board height and width from 2 to 26 and then taken to a creation menu. In this an empty board of their specified dimensions is displayed with an addendum of extra squares below containing all the regular pieces as well any custom ones they have defined. The user can click on a piece to select it and then if they click anywhere on the board the piece is placed there. They can click on a placed piece to remove it. There is a tick button in between the board and the addendum which saves the current board state when clicked. If a win condition involving royals has been selected and there isn't at least one royal piece for both players, the tick button does not allow the user to continue and outputs a message informing them that their ruleset is currently unplayable. It does the same thing if the user tries to start a piece in check.

8) All of the necessary data is saved to a file.

9) Selecting to delete a ruleset presents the user with all the existing rulesets and allows them to delete one of their choosing.

10) Selecting to play a custom ruleset presents the user with all the existing rulesets and allows the user to play one of their choosing.

11）Once a game has ended, if any key or button is pressed the window closes.

12) Before a game is started the user is asked whether they want to record the game. If the user assents, then every time a move is made the letter of the piece is printed and the square which they moved to is printed in algebraic chess notation.

13) The squares of the board and pieces change size based on the user specified height and width so that everything fits on screen at once. The height and width of the user's monitor are used to ensure that the board is as large as possible.

14) If the user types "quit" in the main menu prompt the code terminates.

15) If the user types "help", a help menu is displayed.

Omissions

Special moves

I have chosen to omit the rules of 'en passant' and 'castling' as the focus of my program is on variations of conventional chess and these rules only make sense with the exact board and piece setup of conventional chess. Similar rules could exist for variants, but they would have to be tailored specifically to each variant. Considering the fact that it took hundreds of years for these rules to be decided and

they were not even universally accepted, in the case of conventional chess, it seemed more fitting to remove them entirely as this does not detract from the core elements of the game.

Value of pieces

In chess all pieces have an assigned point value. These values are used to gauge which player is ahead based on the pieces they have remaining. This is called point difference. Most chess applications display this information throughout the game. I have decided to omit point values entirely for three reasons.

Firstly, point difference is not a good measure of advantage because it doesn't factor in where on the board pieces are and what other pieces they are playing with or against. Consideration of these positional factors is a necessary skill for improving at chess and reliance on point difference obstructs this skill's development.

Secondly, the assigned point values are specific to a conventional chess board. They are not based purely on the piece's movement and have been arrived at after much consideration of the piece's value within the context of conventional chess. This includes: the board they are played on, the other pieces they play alongside and against, the goal of the game etc. The value of pieces was established after much scholarly debate and a ranking system was only universally agreed upon after AI mastery of chess disproved the common opinion. However, once you alter the conditions of the game, these piece values become irrelevant. On larger boards, riders have a significant increase in the number of moves they can make increasing their value, but, on boards with a high piece density, pieces with higher movement vectors can navigate more easily by jumping over their friendly pieces and penetrating deeper into enemy ranks. This adds another layer of complexity to the game because, when faced with a new variant, players must work out for themselves how valuable their pieces are and what trades are favourable. The focus of my project is the variations of chess rather than regular chess, so, since the standard piece values only apply in regular chess, there seemed little point in adding them to the game.

Acceptable limitations

Minimum Board Constraints

For Fischer random+ chess, the minimum board height and width is 4. This is because there need to be 4 ranks to contain all the necessary pieces. If the board size is any less than 4 by 4, pieces are likely to start in check.

Maximum Board Constraints

The maximum board size is 26 because in algebraic chess notation (see requirement 12) the letters of the alphabet are used to denote the file of the square being referenced so it is impossible to carry on this convention past 26 letters. On top of this the pieces decrease in size as the board increases so at a certain point, they become difficult to distinguish.

Custom Piece Limit

A maximum of three pieces can be defined by the user. To have an infinite number of pieces generated, there would need to be a chess piece generating algorithm or the custom pieces would all have to

represented by something that appears out of place on a chessboard for example different coloured regular polygons. Since I decided not to make an infinite number of pieces, I selected an arbitrary number of pieces to be the maximum, hence three.

Playability Verification

In board design, the program prevents the user from saving their ruleset if it has a piece in check or if one side doesn't have a piece needed to satisfy a win condition, however, it is still technically possible for a ruleset to be created in which neither player can win. For example, a ruleset where there is a white king, a black king and nothing else. Although there are kings to be checkmated, it is impossible for either side to win. There are two possible approaches to truly verify whether a ruleset is winnable and won't lead to a situation where neither player can make progress. One is using a heuristic set of conditions to work out if checkmate is possible. For conventional chess, what combinations of pieces can be used to win is well documented, for example it is has been mathematically proven that a king and a rook can always checkmate just a king but that a king and two knights can't. A database of these minimum requirements could be cross referenced with the contents of the board to ensure either side can win. However, these are not hard and fast rules and there are certain scenarios where these rules do not hold true. For example, a smothered mate can be achieved with just a knight. If this heuristic ruleset was applied, boundary cases where checkmate is possible could be banned. This doesn't even account for the addition of custom move pieces with undocumented move patterns, in which case a heuristic method would be useless. The only other approach would be to use an AI that plays the variant in order to figure out whether it is winnable. Not only would this method require a significant amount of testing to know if it works, but it might have to play multiple games each time to find a winning series of moves. Variants of chess can have a much higher game complexity than chess. In terms of board size, the maximum board possible is 26 by 26 which is over 10 times as large as a normal chess board which will cause a significant increase in complexity. This is not a worthwhile solution to pursue because of the time complexity that would be required which would slow down other processes. This is significant because speed of play and design is an important part of the user experience.

Promotion display

When promoting a pawn, ideally a window would appear displaying all the pieces for the user to select from. However, this is very difficult to achieve in pygame without multiprocessing, so instead I use a text-based interface that achieves the same thing.