AlphaDeepChess: chess engine based on alpha-beta pruning

Final Degree Project
Course 2024–2025

Authors
Juan Girón Herranz
Yi Wang Qiu

Directors
Ignacio Fábregas Álfaro
Rubén Rafael Rubio Cuéllar

Bachelor's Degree in Computer Engineering
Faculty of Computer Science
Complutense University of Madrid

Bachelor's Degree in Videogame Development
Faculty of Computer Science
Complutense University of Madrid

AlphaDeepChess: chess engine based on alpha-beta pruning

Final Degree Project in Computer Engineering
Final Degree Project in Videogame Development

Authors
Juan Girón Herranz
Yi Wang Qiu

Directors
Ignacio Fábregas Álfaro
Rubén Rafael Rubio Cuéllar

Convocation: *June 2025*

Bachelor's Degree in Computer Engineering
Faculty of Computer Science
Complutense University of Madrid

Bachelor's Degree in Videogame Development
Faculty of Computer Science
Complutense University of Madrid

March 26, 2025

Dedicatory

To our younger selves, for knowing the art of chess

Acknowledgments

To our family members for their support and for taking us to chess tournaments to compete.

Abstract

AlphaDeepChess: chess engine based on alpha-beta pruning

Chess engines have significantly influenced the development of computational strategies and game-playing algorithms since the mid-20th century. Computer scientists as renowned as Alan Turing and Claude Shannon set the foundations for the development of the field. Thereafter, hardware and software improvements and the evolution of heuristics would build upon these foundations, including the introduction of alpha-beta pruning, an optimization of the minimax algorithm that significantly reduced the number of nodes evaluated in a game tree. With increasing computational power, modern engines such as Stockfish or Komodo leverage not only search optimizations but also advancements in heuristics and, in some cases, artificial intelligence using neuronal networks.

Keywords

chess engine,alpha-beta pruning,minimax algorithm,game tree search,heuristic evaluation,move ordering,search optimization,transposition tables,zobrist hashing

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"The most powerful weapon in chess is to have the next move"
— David Bronstein

Chess, one of the oldest and most strategic games in human history, has long been a domain for both intellectual competition and computational research. The pursuit of creating a machine that could compete with the best human players, chess Grandmasters (GM), was present. It was only a matter of time before computation surpassed human computational capabilities.

In 1997, the chess engine Deep Blue made history by defeating the world champion at the time, Garry Kasparov, marking the first time a computer had defeated a reigning world champion in a six-game match under standard chess tournament time controls[1].

Since then, the development of chess engines has advanced rapidly, moving from rule-based systems to AI-driven models. However, classical search algorithms, such as alpha-beta pruning, continue to be fundamental to understanding the basics of efficient search and evaluation of game trees.

1.1 Objetives

- Develop a functional chess engine using alpha-beta pruning as the core search algorithm.
- Optimize search efficiency by implementing move ordering, quiescence search, and iterative deepening to improve pruning effectiveness.
- Implement transposition tables using Zobrist hashing to store and retrieve previously evaluated board positions efficiently.

[1] `https://en.wikipedia.org/wiki/Deep_Blue_(chess_computer)`

- Ensure modularity and efficiency so that the engine can be tested, improved, and integrated into chess-playing applications.
- Compare performance metrics against other classical engines to evaluate the impact of implemented optimizations.

1.2 Work plan

1. Research phase and basic implementation: understand the fundamentals of alpha-beta pruning with minimax and position evaluation. Familiarize with the UCI (Universal Chess Interface) and implement the move generator with its specific exceptions and rules.
2. Optimization: improve search efficiency using transposition tables and Zobrist hashing.
3. Comparation: use Stockfish to compare efficiency generating tournaments between chess engines and a profiler to detect possible bottlenecks.

Chapter 2

Status of the art

Chapter 3

Work description

3.1 Where to begin?

Let's start from the beginning. What is chess? Chess is a board game where two players who take white pieces and black pieces respectively compete to first checkmate[1] the opponent.

What about a chess engine? A chess engine consists of a software program that analyzes chess positions and returns optimal moves depending on its configuration. In order to help users to use these engines, chess community agreed on creating an open communication protocol called **Universal Chess Interface** or commonly referred to as UCI, that provides the interaction with chess engines through user interfaces.

In the following section 3.2, we will talk about the basic concepts of chess, but if you already have the knowledge we recommend you to advance directly to the section 3.3.

3.2 Basic concepts

Chess is a game of strategy that takes place on a chessboard with specific rules governing the movement and interaction of the pieces. This section introduces the fundamental concepts necessary to understand how chess is played.

3.2.1 Chessboard

A chessboard is a game board of 64 squares, 8 rows by 8 columns. To refer to each of the squares we mostly use **algebraic notation**[2] using the numbers from 1 to 8 and the letters from "a" to "h". There are also other notations like descriptive

[1] `https://en.wikipedia.org/wiki/Checkmate`
[2] `https://en.wikipedia.org/wiki/Algebraic_notation_(chess)`

notation[3] which is obsolete or ICCF numeric notation[4] due to chess pieces have different abbreviations depending on language.

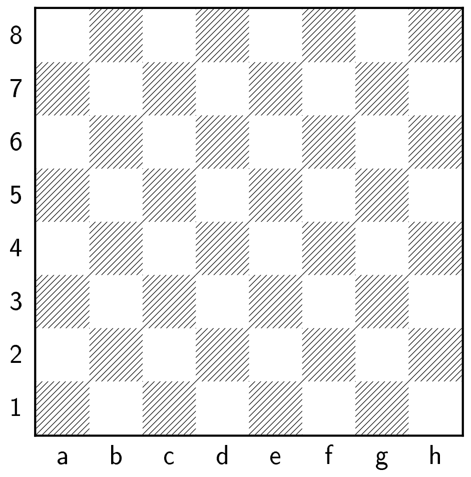

Figure 3.1: Empty chessboard.

For example, **g5** refers to the following square:

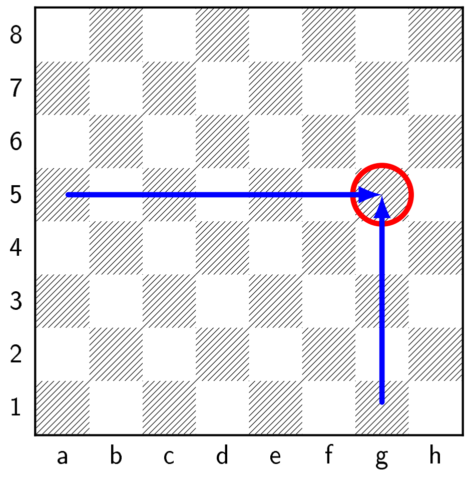

Figure 3.2: Example: square **g5** highlighted and arrows pointing to it.

It is important to know that when placing a chessboard in the correct orientation, there should always be a **white square in the bottom-right corner** or a **black square in the bottom-left corner**.

[3]https://en.wikipedia.org/wiki/Descriptive_notation
[4]https://en.wikipedia.org/wiki/ICCF_numeric_notation

3.2.2 Chess pieces

There are 6 types of chess pieces: king, queen, rook, bishop, knight and pawn, and each side has 16 pieces:

- 1 king ♔♚
- 1 queen ♕♛
- 2 rooks ♖♖♜♜
- 2 bishops ♗♗♝♝
- 2 knights ♘♘♞♞
- 8 pawns ♙♙♙♙♙♙♙♙♟♟♟♟♟♟♟♟

The starting position of the chess pieces on a chessboard is the following:

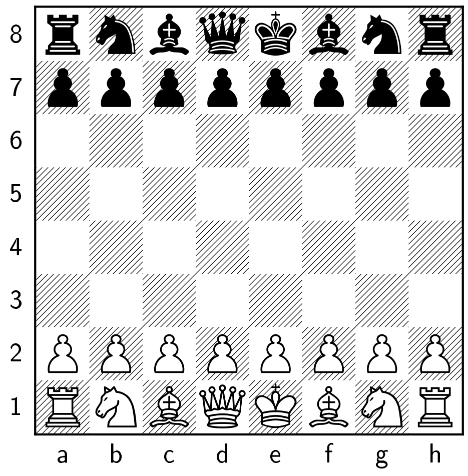

Figure 3.3: Starting position.

If we take a closer look to the order of positioning of each piece, we should consider that queen and king are placed in the center columns. The queen must be placed on a square of their color and the king is placed on the remaining central column. The rest of the pieces are placed symmetrically as seen in figure 3.3.

3.2.3 Movement of the pieces

3.2.4 Notation

3.2.5 Rules

White starts first, possible results (win, draw or lose), 30 moves rule, zugzwang?, check, checkmate, ...

3.3 Modules

3.3.1 Board

3.3.2 Move generator

3.3.3 Move ordering

3.3.4 Evaluation

3.3.5 Search

3.4 Code implementation

3.4.1 Data representation

Use of uint64_t as bitboards to store the board information and other code structures and classes justifying why is efficient.

3.4.2 Initialized memory

Some tables are memory initialized instead of computed, explain it.

3.5 Other

3.5.1 Board visualizer using Python

3.5.2 Comparation using Cutechess and Stockfish engine

3.5.3 Profiling

Chapter 4

Conclusions and Future Work

Conclusiones del trabajo y líneas de trabajo futuro.

Antes de la entrega de actas de cada convocatoria, en el plazo que se indica en el calendario de los trabajos de fin de grado, el estudiante entregará en el Campus Virtual la versión final de la memoria en PDF.

Personal contributions

Student1

Al menos dos páginas con las contribuciones del estudiante 1.

Student 2

Al menos dos páginas con las contribuciones del estudiante 2.

Appendix A

Título del Apéndice A

Los apéndices son secciones al final del documento en las que se agrega texto con el objetivo de ampliar los contenidos del documento principal.

Appendix B

Título del Apéndice B

Se pueden añadir los apéndices que se consideren oportunos.

Este texto se puede encontrar en el fichero Cascaras/fin.tex. Si deseas eliminarlo, basta con comentar la línea correspondiente al final del fichero TFGTeXiS.tex.

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes